

Dipper in NIMBLE

Iraida Redondo & Ana Payo-Payo

21/12/2021

Dipper CMR in NIMBLE

Hi! This RMarkdown document contains the code to run CJS models on the famous dipper dataset in NIMBLE (De Valpine et al. 2017). This code, as well as most of its comments, is founded on the scripts from the wonderful workshop titled **“Bayesian capture-recapture inference with hidden Markov models”** (<https://oliviergimenez.github.io/bayesian-cr-workshop/>) taught by O. Gimenez, C. R. Nater, S. Cubaynes, P. de Valpine, M. QuÄrouÄ© (which I highly recommend if you are interested in conducting capture-recapture analysis in the Bayesian framework). The dipper dataset can be found in <https://oliviergimenez.github.io/bayesian-cr-workshop/> in the Live Demos tab on the upper right side of the website. Within this zip you can find the dipper dataset in form of a .csv.

This script contains a great variety of models. This has served me as insightful exercise to know how to specify different models and to check if I was doing it right for my own data analysis. I think most of its content is right but I am currently learning and the code is very long, so there may be errors! Please, don't doubt and contact me if you spot any! MCMC configuration is free to be changed!

Libraries and setting wordking directory

```
library(nimble)
library(tidyverse)
library(MCMCvis)

setwd() # depends on the person running the script!
```

Loading dipper dataset.

```
dipper_d <- read_csv("dipper.csv") # data
sex <- ifelse(dipper_d$sex=="Male",1 , 2) # vector for sex: 1 = males, 2 = females

# Format data
y <- dipper_d %>%
  select(year_1981:year_1987) %>%
  as.matrix()
head(y)
```

Cormack-Jolly-Seber (CJS) models

Survival as constant $\sim \phi(\cdot)$

$\phi(\cdot)$ $p(\cdot)$

```
##### phi(.)p(.)-----
hmm.phip <- nimbleCode({

  #Initial state prob.
  delta[1] <- 1      # Pr(alive t = 1) = 1
  delta[2] <- 0      # Pr(dead t = 1) = 0

  #Survival
  phi ~ dunif(0, 1) # prior survival
  #Survival matrix
  gamma[1,1] <- phi   # Pr(alive t -> alive t+1)
  gamma[1,2] <- 1 - phi # Pr(alive t -> dead t+1)
  gamma[2,1] <- 0     # Pr(dead t -> alive t+1)
  gamma[2,2] <- 1     # Pr(dead t -> dead t+1)

  #Recapture
  p ~ dunif(0, 1) # prior detection
  #Recapture matrix
  omega[1,1] <- 1 - p # Pr(alive t -> non-detected t)
  omega[1,2] <- p     # Pr(alive t -> detected t)
  omega[2,1] <- 1     # Pr(dead t -> non-detected t)
  omega[2,2] <- 0     # Pr(dead t -> detected t)

  #Likelihood
  for (i in 1:N){
    z[i,first[i]] ~ dcat(delta[1:2])
    for (j in (first[i]+1):T){
      z[i,j] ~ dcat(gamma[z[i,j-1], 1:2])
      y[i,j] ~ dcat(omega[z[i,j], 1:2])
    }
  }
})

#Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first)
my.constants

#Now the data in a list. Note that we add 1 to the data
```

```

#to have 1 for non-detections and 2 for detections.
#You may use the coding you prefer of course, you will just need to
#adjust the  $\Omega$  and  $\Gamma$  matrices in the model above.
my.data <- list(y = y + 1)

#Specify initial values. For the latent states, we go for the easy way,
#and say that all individuals are alive through the study period.
zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(phi = runif(1,0,1),
                                   p = runif(1,0,1),
                                   z = zinits)

initial.values()

#Some information that we now pass as initial value
#info (observations of alive) are actually known states,
#and could also be passed as data in which case the initial values have to be 0.
#Specify the parameters we wish to monitor.
parameters.to.save <- c("phi", "p")
parameters.to.save

# MCMC details
n.iter <- 2500
n.burnin <- 1000
n.chains <- 2

#Let's run nimble.
mcmc.phip <- nimbleMCMC(code = hmm.phip,
                        constants = my.constants,
                        data = my.data,
                        inits = initial.values,
                        monitors = parameters.to.save,
                        niter = n.iter,
                        nburnin = n.burnin,
                        nchains = n.chains)

#Examine the results.
MCMCsummary(mcmc.phip, round = 2)
MCMCtrace(mcmc.phip, pdf = F)

```

$\phi(\cdot)$ $p(\text{sex})$

```

#### phi(.)p(s)-----
hmm.phips <- nimbleCode({

  #Initial state prob.
  delta[1] <- 1          # Pr(alive t = 1) = 1
  delta[2] <- 0          # Pr(dead t = 1) = 0

```

```

#Survival
phi ~ dunif(0,1) # prior survival
#Survival matrix
gamma[1,1] <- phi      # Pr(alive t -> alive t+1)
gamma[1,2] <- 1 - phi  # Pr(alive t -> dead t+1)
gamma[2,1] <- 0        # Pr(dead t -> alive t+1)
gamma[2,2] <- 1        # Pr(dead t -> dead t+1)

#Recapture depends on sex
for(i in 1:N){
logit(p[i]) <- beta[sex[i]]
#Observation matrix
omega[1,1,i] <- 1 - p[i]  # Pr(alive t -> non-detected t)
omega[1,2,i] <- p[i]      # Pr(alive t -> detected t)
omega[2,1,i] <- 1         # Pr(dead t -> non-detected t)
omega[2,2,i] <- 0         # Pr(dead t -> detected t)
}

# Priors for beta (recapture changes with sex, so we need two betas;
#beta[sex[i] -> beta[1] and beta[2]])
beta[1] ~ dnorm(mean = 0, sd = 1.5)
beta[2] ~ dnorm(mean = 0, sd = 1.5)

# inverse logit for transforming p estimate
p_male <- ilogit(beta[1])
p_female <- ilogit(beta[2])

#Likelihood
for (i in 1:N){
  z[i,first[i]] ~ dcat(delta[1:2])
  for (j in (first[i]+1):T){
    z[i,j] ~ dcat(gamma[z[i,j-1], 1:2])
    y[i,j] ~ dcat(omega[z[i,j], 1:2, i])
  }
}
})

#Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)

#Data in a list. We add 1 to the data to have 1 for non-detections and 2 for detections.
my.data <- list(y = y + 1)

#Initial values

```

```

zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                   phi = runif(1,0,1),
                                   z = zinits)

initial.values()

#Specify the parameters we wish to monitor.
parameters.to.save <- c("beta", "phi", "p_male", "p_female")
parameters.to.save

#MCMC details.
n.iter <- 2500
n.burnin <- 1000
n.chains <- 2

#At last, let's run nimble.
mcmc.phips <- nimbleMCMC(code = hmm.phips,
                        constants = my.constants,
                        data = my.data,
                        inits = initial.values,
                        monitors = parameters.to.save,
                        niter = n.iter,
                        nburnin = n.burnin,
                        nchains = n.chains)

#' Examine the results.
MCMCsummary(mcmc.phips, round = 2)
MCMCtrace(mcmc.phips, pdf=F)

```

$\phi(\cdot)$ $p(t)$

```

##### phi(.)p(t)-----
hmm.phipt <- nimbleCode({

  #Initial state prob.
  delta[1] <- 1      # Pr(alive t = 1) = 1
  delta[2] <- 0      # Pr(dead t = 1) = 0

  #Survival
  phi ~ dunif(0, 1) # Prior for survival
  #Survival matrix
  gamma[1,1] <- phi  # Pr(alive t -> alive t+1)
  gamma[1,2] <- 1 - phi # Pr(alive t -> dead t+1)
  gamma[2,1] <- 0    # Pr(dead t -> alive t+1)
  gamma[2,2] <- 1    # Pr(dead t -> dead t+1)

```

```

#Recapture
for(t in 1:(T-1)){
p[t] ~ dunif(0,1) # Prior for p.
#Recapture matrix
omega[1,1,t] <- 1 - p[t]      # Pr(alive t -> non-detected t)
omega[1,2,t] <- p[t]         # Pr(alive t -> detected t)
omega[2,1,t] <- 1            # Pr(dead t -> non-detected t)
omega[2,2,t] <- 0            # Pr(dead t -> detected t)
}

#Likelihood
for (i in 1:N){
  z[i,first[i]] ~ dcat(delta[1:2])
  for (j in (first[i]+1):T){
    z[i,j] ~ dcat(gamma[z[i,j-1], 1:2])
    y[i,j] ~ dcat(omega[z[i,j], 1:2, j-1])
  }
}
})

#Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first)
my.constants

#Now the data in a list.
my.data <- list(y = y + 1)

#Initial values.
zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(phi = runif(1,0,1),
                                   p = runif(my.constants$T-1,0,1),
                                   z = zinits)
initial.values()

#Specify the parameters we wish to monitor.
parameters.to.save <- c("phi", "p")
parameters.to.save

#MCMC details.
n.iter <- 8000

```

```

n.burnin <- 1000
n.chains <- 2

#At last, let's run nimble.
mcmc.phipt <- nimbleMCMC(code = hmm.phipt,
                        constants = my.constants,
                        data = my.data,
                        inits = initial.values,
                        monitors = parameters.to.save,
                        niter = n.iter,
                        nburnin = n.burnin,
                        nchains = n.chains)

#' Examine the results.
MCMCsummary(mcmc.phipt, round = 2)
MCMCtrace(mcmc.phipt, pdf=F)

```

$\phi(\cdot)$ $p(\text{sex}+t)$

```

##### phi(.)p(s+t)-----
hmm.phips_t <- nimbleCode({
  #Initial state prob.
  delta[1] <- 1      # Pr(alive t = 1) = 1
  delta[2] <- 0      # Pr(dead t = 1) = 0

  #Survival
  phi ~ dunif(0,1) # prior survival
  #Survival matrix
  gamma[1,1] <- phi   # Pr(alive t -> alive t+1)
  gamma[1,2] <- 1 - phi # Pr(alive t -> dead t+1)
  gamma[2,1] <- 0     # Pr(dead t -> alive t+1)
  gamma[2,2] <- 1     # Pr(dead t -> dead t+1)

  # Recapture
  for(i in 1:N){
    for(t in 1:(T-1)){
      logit(p[i,t]) <- beta[sex[i]]+ lambda[t] #additive time + sex
      #Recapture matrix
      omega[1,1,i,t] <- 1 - p[i,t] # Pr(alive t -> non-detected t)
      omega[1,2,i,t] <- p[i,t]    # Pr(alive t -> detected t)
      omega[2,1,i,t] <- 1         # Pr(dead t -> non-detected t)
      omega[2,2,i,t] <- 0         # Pr(dead t -> detected t)
    }
  }

  #Priors for beta
  beta[1] ~ dnorm(mean = 0, sd = 1.5)
  beta[2] ~ dnorm(mean = 0, sd = 1.5)

```

```

# Time fixed effect.
for(t in 1:(T-1)){
  lambda[t] ~ dnorm(0, sd = 1.5)
}

# ilogit for p.
for (t in 1:(T-1)){
  p_male[t] <- ilogit(beta[1]+ lambda[t])
  p_female[t] <- ilogit(beta[2] + lambda[t])
}

#Likelihood
for (i in 1:N){
  z[i,first[i]] ~ dcat(delta[1:2])
  for (j in (first[i]+1):T){
    z[i,j] ~ dcat(gamma[z[i,j-1], 1:2])
    y[i,j] ~ dcat(omega[z[i,j], 1:2, i, j-1])
  }
}
})

#Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)
my.constants

#Now the data in a list.
my.data <- list(y = y + 1)

#Specify initial values.
zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                   phi = runif(1,0,1),
                                   lambda = rnorm(my.constants$T-1, 0, 1),
                                   z = zinits)
initial.values()

#Specify the parameters we wish to monitor.
parameters.to.save <- c("phi", "p_male", "p_female")
parameters.to.save

```



```

#MCMC details.
n.iter <- 15000
n.burnin <- 5000
n.chains <- 2

#At last, let's run nimble.
mcmc.phips_t <- nimbleMCMC(code = hmm.phips_t,
                           constants = my.constants,
                           data = my.data,
                           inits = initial.values,
                           monitors = parameters.to.save,
                           niter = n.iter,
                           nburnin = n.burnin,
                           nchains = n.chains)

#Examine the results.
MCMCsummary(mcmc.phips_t, round = 2)
MCMCtrace(mcmc.phips_t, pdf=F)

```

$\phi(\cdot)$ $p(\text{sex} \cdot t)$

```

##### phi(.)p(s*t)-----
hmm.phipst <- nimbleCode({

  #Initial state prob.
  delta[1] <- 1      # Pr(alive t = 1) = 1
  delta[2] <- 0      # Pr(dead t = 1) = 0

  #Survival
  phi ~ dunif(0,1) # prior survival
  #Survival matrix
  gamma[1,1] <- phi   # Pr(alive t -> alive t+1)
  gamma[1,2] <- 1 - phi # Pr(alive t -> dead t+1)
  gamma[2,1] <- 0     # Pr(dead t -> alive t+1)
  gamma[2,2] <- 1     # Pr(dead t -> dead t+1)

  # Recapture
  for(i in 1:N){
    for(t in 1:(T-1)){
      logit(p[i,t]) <- beta[sex[i]] + lambda[t] + kappa[sex[i],t] #interaction sex * time
      #Recapture matrix
      omega[1,1,i,t] <- 1 - p[i,t]   # Pr(alive t -> non-detected t)
      omega[1,2,i,t] <- p[i,t]       # Pr(alive t -> detected t)
      omega[2,1,i,t] <- 1             # Pr(dead t -> non-detected t)
      omega[2,2,i,t] <- 0             # Pr(dead t -> detected t)
    }
  }

  #Priors for beta
  beta[1] ~ dnorm(mean = 0, sd = 1.5)

```

```

beta[2] ~ dnorm(mean = 0, sd = 1.5)

#Time fixed effect.
for(t in 1:(T-1)){
  lambda[t] ~ dnorm(0, sd = 1.5)
}

#Time as random effect in the interaction
lambda.sigma ~ dunif(0, 10)
for(i in 1:2){
  for (t in 1:(T-1)){
    kappa[i,t] ~ dnorm(0, sd = lambda.sigma)
  }
}

# ilogit for p.
for (t in 1:(T-1)){
  p_male[t] <- ilogit(beta[1]+ lambda[t] + kappa[1,t])
  p_female[t] <- ilogit(beta[2] + lambda[t] + kappa[2,t])
}

#Likelihood
for (i in 1:N){
  z[i,first[i]] ~ dcat(delta[1:2])
  for (j in (first[i]+1):T){
    z[i,j] ~ dcat(gamma[z[i,j-1], 1:2])
    y[i,j] ~ dcat(omega[z[i,j], 1:2, i, j-1])
  }
}
})

#Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)
my.constants

#Now the data in a list.
my.data <- list(y = y + 1)

#Specify initial values
zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                   phi = runif(1,0,1),

```

```

        lambda = rnorm(my.constants$T-1, 0, 1),
        lambda.sigma = runif(1,0,1),
        kappa = matrix(rnorm(12, 0, 1), 2, 6),
        z = zinits)
initial.values() #note kappa is a 2-row matrix: [1,] -> male; [2,] -> female
               # with number columns = time occasions.

#Specify the parameters we wish to monitor.
parameters.to.save <- c("phi", "p_male", "p_female")
parameters.to.save

#MCMC details.
n.iter <- 15000
n.burnin <- 5000
n.chains <- 2

#At last, let's run nimble.
mcmc.phipst <- nimbleMCMC(code = hmm.phipst,
                        constants = my.constants,
                        data = my.data,
                        inits = initial.values,
                        monitors = parameters.to.save,
                        niter = n.iter,
                        nburnin = n.burnin,
                        nchains = n.chains)

#Examine the results.
MCMCsummary(mcmc.phipst, round = 2)
MCMCtrace(mcmc.phipst,pdf=F)

```

Survival dependent on time $\sim \phi(t)$

$\phi(t)$ $p(\cdot)$

```

##### phi(t)p(.)-----
hmm.phitp <- nimbleCode({

  #Initial state prob.
  delta[1] <- 1          # Pr(alive t = 1) = 1
  delta[2] <- 0          # Pr(dead t = 1) = 0

  # Survival
  for(t in 1:(T-1)){
    phi[t] ~ dunif(0,1) # prior survival
    #Survival matrix
    gamma[1,1,t] <- phi[t]      # Pr(alive t -> alive t+1)
    gamma[1,2,t] <- 1 - phi[t]  # Pr(alive t -> dead t+1)
    gamma[2,1,t] <- 0          # Pr(dead t -> alive t+1)
    gamma[2,2,t] <- 1          # Pr(dead t -> dead t+1)
  }

```

```

}

#Recapture matrix
p ~ dunif(0, 1) # prior detection
omega[1,1] <- 1 - p      # Pr(alive t -> non-detected t)
omega[1,2] <- p          # Pr(alive t -> detected t)
omega[2,1] <- 1          # Pr(dead t -> non-detected t)
omega[2,2] <- 0          # Pr(dead t -> detected t)

#Likelihood
for (i in 1:N){
  z[i,first[i]] ~ dcat(delta[1:2])
  for (j in (first[i]+1):T){
    z[i,j] ~ dcat(gamma[z[i,j-1], 1:2, j-1])
    y[i,j] ~ dcat(omega[z[i,j], 1:2])
  }
}
})

#Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first)
my.constants

#Now the data in a list.
my.data <- list(y = y + 1)

#Specify initial values.
zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(phi = runif(my.constants$T-1,0,1),
                                   p = runif(1,0,1),
                                   z = zinits)
initial.values()

#Specify the parameters we wish to monitor.
parameters.to.save <- c("phi", "p")
parameters.to.save

#MCMC details.
n.iter <- 2500
n.burnin <- 1000
n.chains <- 2

```

```

#At last, let's run nimble.
mcmc.phitp <- nimbleMCMC(code = hmm.phitp,
                        constants = my.constants,
                        data = my.data,
                        inits = initial.values,
                        monitors = parameters.to.save,
                        niter = n.iter,
                        nburnin = n.burnin,
                        nchains = n.chains)

#Examine the results.
MCMCsummary(mcmc.phitp, round = 2)
MCMCtrace(mcmc.phitp, params = "p", pdf=F)

```

```

##### phi(t)p(s)-----
hmm.phitps <- nimbleCode({

  #Initial state prob.
  delta[1] <- 1      # Pr(alive t = 1) = 1
  delta[2] <- 0      # Pr(dead t = 1) = 0

  #Survival
  for(t in 1:(T-1)){
    phi[t] ~ dunif(0,1) # prior for phi
    #Survival matrix
    gamma[1,1,t] <- phi[t]      # Pr(alive t -> alive t+1)
    gamma[1,2,t] <- 1 - phi[t]  # Pr(alive t -> dead t+1)
    gamma[2,1,t] <- 0           # Pr(dead t -> alive t+1)
    gamma[2,2,t] <- 1           # Pr(dead t -> dead t+1)
  }

  #Recapture
  for(i in 1:N){
    logit(p[i]) <- beta[sex[i]]
    #Observation matrix
    omega[1,1,i] <- 1 - p[i]    # Pr(alive t -> non-detected t)
    omega[1,2,i] <- p[i]       # Pr(alive t -> detected t)
    omega[2,1,i] <- 1          # Pr(dead t -> non-detected t)
    omega[2,2,i] <- 0          # Pr(dead t -> detected t)
  }

  #Priors for beta
  beta[1] ~ dnorm(mean = 0, sd = 1.5)
  beta[2] ~ dnorm(mean = 0, sd = 1.5)

  # ilogit for p
  p_male <- ilogit(beta[1])
  p_female <- ilogit(beta[2])

  # Likelihood

```

```

    for (i in 1:N){
      z[i,first[i]] ~ dcat(delta[1:2])
      for (j in (first[i]+1):T){
        z[i,j] ~ dcat(gamma[z[i,j-1], 1:2, j-1])
        y[i,j] ~ dcat(omega[z[i,j], 1:2, i])
      }
    }
  })

#Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)
my.constants

#Now the data in a list.
my.data <- list(y = y + 1)

#Specify initial values.
zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(phi = runif(my.constants$T-1,0,1),
                                   beta = rnorm(2,0,1),
                                   z = zinits)
initial.values()

#Specify the parameters we wish to monitor.
parameters.to.save <- c("phi", "p_male", "p_female", "beta")
parameters.to.save

#MCMC details.
n.iter <- 2500
n.burnin <- 1000
n.chains <- 2

#At last, let's run nimble.
mcmc.phitps <- nimbleMCMC(code = hmm.phitps,
                         constants = my.constants,
                         data = my.data,
                         inits = initial.values,
                         monitors = parameters.to.save,

```

```

niter = n.iter,
nburnin = n.burnin,
nchains = n.chains)

```

#Examine the results.

```

MCMCsummary(mcmc.phitps, round = 2)
MCMCtrace(mcmc.phitps, params = "all",pdf=F)

```

$\phi(t)$ $p(\text{sex})$

$\phi(t)$ $p(t)$

```

##### phi(t)p(t)-----
hmm.phitpt <- nimbleCode({

  #Initial state prob
  delta[1] <- 1      # Pr(alive t = 1) = 1
  delta[2] <- 0      # Pr(dead t = 1) = 0

  #Survival
  for(t in 1:(T-1)){
    phi[t] ~ dunif(0,1) # prior for phi
    #Survival matrix
    gamma[1,1,t] <- phi[t]      # Pr(alive t -> alive t+1)
    gamma[1,2,t] <- 1 - phi[t]  # Pr(alive t -> dead t+1)
    gamma[2,1,t] <- 0           # Pr(dead t -> alive t+1)
    gamma[2,2,t] <- 1           # Pr(dead t -> dead t+1)
  }

  #Recapture
  for(t in 1:(T-1)){
    p[t] ~ dunif(0,1) # prior for p
    # Recapture matrix
    omega[1,1,t] <- 1 - p[t]    # Pr(alive t -> non-detected t)
    omega[1,2,t] <- p[t]       # Pr(alive t -> detected t)
    omega[2,1,t] <- 1          # Pr(dead t -> non-detected t)
    omega[2,2,t] <- 0          # Pr(dead t -> detected t)
  }

  #Likelihood
  for (i in 1:N){
    z[i,first[i]] ~ dcat(delta[1:2])
    for (j in (first[i]+1):T){
      z[i,j] ~ dcat(gamma[z[i,j-1], 1:2, j-1])
      y[i,j] ~ dcat(omega[z[i,j], 1:2, j-1])
    }
  }
})

#Get the occasion of first capture for all individuals.

```

```

first <- apply(y, 1, function(x) min(which(x !=0)))
first

#A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first)
my.constants

#Now the data in a list. N
my.data <- list(y = y + 1)

#Specify initial values.
zinit <- y + 1 # non-detection -> alive
zinit[zinit == 2] <- 1 # dead -> alive
initial.values <- function() list(phi = runif(my.constants$T-1,0,1),
                                   p = runif(my.constants$T-1,0,1),
                                   z = zinit)

initial.values()

#Specify the parameters we wish to monitor.
parameters.to.save <- c("phi", "p")
parameters.to.save

#MCMC details.
n.iter <- 2500
n.burnin <- 1000
n.chains <- 2

#At last, let's run nimble.
mcmc.phitpt <- nimbleMCMC(code = hmm.phitpt,
                        constants = my.constants,
                        data = my.data,
                        inits = initial.values,
                        monitors = parameters.to.save,
                        niter = n.iter,
                        nburnin = n.burnin,
                        nchains = n.chains)

#Examine the results.
MCMCsummary(mcmc.phitpt, round = 2)
MCMCtrace(mcmc.phitpt, params = "all",pdf=F)

```

$\phi(t)$ $p(\text{sex}+t)$

```

##### phi(t)p(s+t)-----
hmm.phitps_t <- nimbleCode({

```



```

#Initial state prob
delta[1] <- 1          # Pr(alive t = 1) = 1
delta[2] <- 0          # Pr(dead t = 1) = 0

#Survival
for(t in 1:(T-1)){
  phi[t] ~ dunif(0,1) # prior for phi
  #Survival matrix
  gamma[1,1,t] <- phi[t]      # Pr(alive t -> alive t+1)
  gamma[1,2,t] <- 1 - phi[t]  # Pr(alive t -> dead t+1)
  gamma[2,1,t] <- 0           # Pr(dead t -> alive t+1)
  gamma[2,2,t] <- 1           # Pr(dead t -> dead t+1)
}

#Recapture
for(i in 1:N){
  for(t in 1:(T-1)){
    logit(p[i,t]) <- beta[sex[i]] + lambda[t]

    #Recapture matrix
    omega[1,1,i,t] <- 1 - p[i,t]  # Pr(alive t -> non-detected t)
    omega[1,2,i,t] <- p[i,t]      # Pr(alive t -> detected t)
    omega[2,1,i,t] <- 1           # Pr(dead t -> non-detected t)
    omega[2,2,i,t] <- 0           # Pr(dead t -> detected t)

  }
}

#Priors for beta
beta[1] ~ dnorm(mean = 0, sd = 1.5)
beta[2] ~ dnorm(mean = 0, sd = 1.5)

# Time fixed effect.
for(t in 1:(T-1)){
  lambda[t] ~ dnorm(0, sd = 1.5)
}

# ilogit for p.
for (t in 1:(T-1)){
  p_male[t] <- ilogit(beta[1]+ lambda[t])
  p_female[t] <- ilogit(beta[2] + lambda[t])
}

#Likelihood
for (i in 1:N){
  z[i,first[i]] ~ dcat(delta[1:2])
  for (j in (first[i]+1):T){
    z[i,j] ~ dcat(gamma[z[i,j-1], 1:2, j-1])
    y[i,j] ~ dcat(omega[z[i,j], 1:2, i, j-1])
  }
}
})

```

```

#Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)

my.constants

#Now the data in a list.
my.data <- list(y = y + 1)

#Specify initial values.
zinit <- y + 1 # non-detection -> alive
zinit[zinit == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                   phi = runif(my.constants$T-1,0,1),
                                   lambda = rnorm(my.constants$T-1, 0, 1),
                                   z = zinit)

initial.values()

#Specify the parameters we wish to monitor.
parameters.to.save <- c("phi", "p_male", "p_female")
parameters.to.save

#MCMC details.
n.iter <- 8000
n.burnin <- 1000
n.chains <- 2

#At last, let's run nimble.
mcmc.phitps_t <- nimbleMCMC(code = hmm.phitps_t,
                           constants = my.constants,
                           data = my.data,
                           inits = initial.values,
                           monitors = parameters.to.save,
                           niter = n.iter,
                           nburnin = n.burnin,
                           nchains = n.chains)

#Examine the results.
MCMCsummary(mcmc.phitps_t, round = 2)
MCMCtrace(mcmc.phitps_t, params = "p_female", pdf=F)

```

$\phi(t)$ $p(\text{sex} \cdot \text{time})$

```
##### phi(t)p(s*t)-----
hmm.phitpst <- nimbleCode({

  #Initial state prob
  delta[1] <- 1          # Pr(alive t = 1) = 1
  delta[2] <- 0          # Pr(dead t = 1) = 0

  #Survival
  for(t in 1:(T-1)){
    phi[t] ~ dunif(0,1) # prior for phi
    #Survival matrix
    gamma[1,1,t] <- phi[t]      # Pr(alive t -> alive t+1)
    gamma[1,2,t] <- 1 - phi[t]  # Pr(alive t -> dead t+1)
    gamma[2,1,t] <- 0           # Pr(dead t -> alive t+1)
    gamma[2,2,t] <- 1           # Pr(dead t -> dead t+1)
  }

  #Recapture
  for(i in 1:N){
    for(t in 1:(T-1)){
      logit(p[i,t]) <- beta[sex[i]] + lambda[t] + kappa[sex[i],t]
      #Recapture matrix
      omega[1,1,i,t] <- 1 - p[i,t]      # Pr(alive t -> non-detected t)
      omega[1,2,i,t] <- p[i,t]          # Pr(alive t -> detected t)
      omega[2,1,i,t] <- 1                # Pr(dead t -> non-detected t)
      omega[2,2,i,t] <- 0                # Pr(dead t -> detected t)

    }
  }

  #Priors for beta
  beta[1] ~ dnorm(mean = 0, sd = 1.5)
  beta[2] ~ dnorm(mean = 0, sd = 1.5)

  # Time fixed effect.
  for(t in 1:(T-1)){
    lambda[t] ~ dnorm(0, sd = 1.5)
  }

  # Time as random effect in the interaction
  lambda.sigma ~ dunif(0, 10)
  for(i in 1:2){
    for (t in 1:(T-1)){
      kappa[i,t] ~ dnorm(0, sd = lambda.sigma)
    }
  }

  # ilogit for p.
  for (t in 1:(T-1)){
    p_male[t] <- ilogit(beta[1]+ lambda[t] + kappa[1,t])
    p_female[t] <- ilogit(beta[2] + lambda[t] + kappa[2,t])
  }
}
```

```

#Likelihood
for (i in 1:N){
  z[i,first[i]] ~ dcat(delta[1:2])
  for (j in (first[i]+1):T){
    z[i,j] ~ dcat(gamma[z[i,j-1], 1:2, j-1])
    y[i,j] ~ dcat(omega[z[i,j], 1:2, i, j-1])
  }
}
})

#Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)

my.constants

#Now the data in a list.
my.data <- list(y = y + 1)

#Specify initial values.
zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                   phi = runif(my.constants$T-1,0,1),
                                   lambda = rnorm(my.constants$T-1, 0, 1),
                                   t.sigma = runif(1,0,1),
                                   kappa = matrix(rnorm(12, 0, 1), 2, 6),
                                   z = zinits)

initial.values()

#Specify the parameters we wish to monitor.
parameters.to.save <- c("phi", "p_male", "p_female")
parameters.to.save

#MCMC details.
n.iter <- 8000
n.burnin <- 1000
n.chains <- 2

#At last, let's run nimble.
mcmc.phitpst <- nimbleMCMC(code = hmm.phitpst,
                          constants = my.constants,

```

```

        data = my.data,
        inits = initial.values,
        monitors = parameters.to.save,
        niter = n.iter,
        nburnin = n.burnin,
        nchains = n.chains)

#Examine the results.
MCMCsummary(mcmc.phitpst, round = 2)
MCMCtrace(mcmc.phitpst, params = "p_female", pdf=F)

```

Survival dependent on sex $\sim \phi(s)$

$\phi(\text{sex}) p(\cdot)$

```

## PHI(s) -----
##### phi(s)p(.)-----

hmm.phisp <- nimbleCode({

  # Initial state prob.
  delta[1] <- 1      # Pr(alive t = 1) = 1
  delta[2] <- 0      # Pr(dead t = 1) = 0

  #Survival
  for(i in 1:N){
    logit(phi[i])<- beta[sex[i]]
    #Survival matrix
    gamma[1,1,i] <- phi[i]      # Pr(alive t -> alive t+1)
    gamma[1,2,i] <- 1 - phi[i]  # Pr(alive t -> dead t+1)
    gamma[2,1,i] <- 0           # Pr(dead t -> alive t+1)
    gamma[2,2,i] <- 1           # Pr(dead t -> dead t+1)
  }

  # Priors for beta
  beta[1] ~ dnorm(mean = 0, sd = 1.5)
  beta[2] ~ dnorm(mean = 0, sd = 1.5)

  # ilogit for phi
  phi_male <- ilogit(beta[1])
  phi_female <- ilogit(beta[2])

  #Recapture
  p ~ dunif(0,1) # prior for p
  # Recapture matrix
  omega[1,1] <- 1 - p      # Pr(alive t -> non-detected t)
  omega[1,2] <- p          # Pr(alive t -> detected t)
  omega[2,1] <- 1          # Pr(dead t -> non-detected t)
  omega[2,2] <- 0          # Pr(dead t -> detected t)

  #Likelihood
  for (i in 1:N){

```

```

    z[i,first[i]] ~ dcat(delta[1:2])
    for (j in (first[i]+1):T){
      z[i,j] ~ dcat(gamma[z[i,j-1], 1:2, i])
      y[i,j] ~ dcat(omega[z[i,j], 1:2])
    }
  }
})

#' Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#' A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)
my.constants

#' Now the data in a list.
my.data <- list(y = y + 1)

#' Specify initial values.
zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                   p = runif(1,0,1),
                                   z = zinits)
initial.values()

#' Specify the parameters we wish to monitor.
parameters.to.save <- c("beta", "phi_male", "phi_female", "p")
parameters.to.save

#' MCMC details.
n.iter <- 2500
n.burnin <- 1000
n.chains <- 2

#' At last, let's run nimble.
mcmc.phisp <- nimbleMCMC(code = hmm.phisp,
                        constants = my.constants,
                        data = my.data,
                        inits = initial.values,
                        monitors = parameters.to.save,
                        niter = n.iter,
                        nburnin = n.burnin,

```

```

nchains = n.chains)

#' Examine the results.
MCMCsummary(mcmc.phisp, round = 2)
MCMCtrace(mcmc.phisp, params = "all", pdf=F)

```

$\phi(\text{sex})$ $p(\text{sex})$

```

##### phi(s)p(s)-----
hmm.phisps <- nimbleCode({

  # Initial state prob
  delta[1] <- 1      # Pr(alive t = 1) = 1
  delta[2] <- 0      # Pr(dead t = 1) = 0

  #Survival
  for(i in 1:N){
    logit(phi[i])<- beta[sex[i]]
    #Survival matrix
    gamma[1,1,i] <- phi[i]      # Pr(alive t -> alive t+1)
    gamma[1,2,i] <- 1 - phi[i]  # Pr(alive t -> dead t+1)
    gamma[2,1,i] <- 0           # Pr(dead t -> alive t+1)
    gamma[2,2,i] <- 1           # Pr(dead t -> dead t+1)
  }

  # Prior for b1
  beta[1] ~ dnorm(mean = 0, sd = 1.5)
  beta[2] ~ dnorm(mean = 0, sd = 1.5)

  #ilogit for phi
  phi_male <- ilogit(beta[1])
  phi_female <- ilogit(beta[2])

  #Recapture
  for(i in 1:N){
    logit(p[i]) <- beta2[sex[i]]
    #Recapture matrix
    omega[1,1,i] <- 1 - p[i]    # Pr(alive t -> non-detected t)
    omega[1,2,i] <- p[i]       # Pr(alive t -> detected t)
    omega[2,1,i] <- 1          # Pr(dead t -> non-detected t)
    omega[2,2,i] <- 0          # Pr(dead t -> detected t)
  }

  # Priors for b2
  beta2[1] ~ dnorm(mean = 0, sd = 1.5)
  beta2[2] ~ dnorm(mean = 0, sd = 1.5)

  #ilogit for p
  p_male <- ilogit(beta2[1])
  p_female <- ilogit(beta2[2])

  # Likelihood

```

```

for (i in 1:N){
  z[i,first[i]] ~ dcat(delta[1:2])
  for (j in (first[i]+1):T){
    z[i,j] ~ dcat(gamma[z[i,j-1], 1:2, i])
    y[i,j] ~ dcat(omega[z[i,j], 1:2, i])
  }
}
})

#' Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#' A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)
my.constants

#' Now the data in a list.
my.data <- list(y = y + 1)

#' Specify initial values.
zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                   beta2 = rnorm(2,0,1),
                                   z = zinits)
initial.values()

#' Specify the parameters we wish to monitor.
parameters.to.save <- c("beta", "phi_male", "phi_female", "p_male", "p_female")
parameters.to.save

#' MCMC details.
n.iter <- 2500
n.burnin <- 1000
n.chains <- 2

#' At last, let's run nimble.
mcmc.phisps <- nimbleMCMC(code = hmm.phisps,
                          constants = my.constants,
                          data = my.data,
                          inits = initial.values,
                          monitors = parameters.to.save,
                          niter = n.iter,
                          nburnin = n.burnin,
                          nchains = n.chains)

```



```
#' Examine the results.
MCMCsummary(mcmc.phisps, round = 2)
MCMCtrace(mcmc.phisps, params = "all", pdf=F)
```

$\phi(\text{sex}) p(t)$

```
##### phi(s)p(t)-----
hmm.phispt <- nimbleCode({

  #Initial state prob.
  delta[1] <- 1      # Pr(alive t = 1) = 1
  delta[2] <- 0      # Pr(dead t = 1) = 0

  #Survival
  for(i in 1:N){
    logit(phi[i])<- beta[sex[i]]
    # Survival matrix
    gamma[1,1,i] <- phi[i]      # Pr(alive t -> alive t+1)
    gamma[1,2,i] <- 1 - phi[i]  # Pr(alive t -> dead t+1)
    gamma[2,1,i] <- 0           # Pr(dead t -> alive t+1)
    gamma[2,2,i] <- 1           # Pr(dead t -> dead t+1)
  }

  # Prior for b1
  beta[1] ~ dnorm(mean = 0, sd = 1.5)
  beta[2] ~ dnorm(mean = 0, sd = 1.5)

  #ilogit for phi
  phi_male <- ilogit(beta[1])
  phi_female <- ilogit(beta[2])

  #Recapture
  for(t in 1:(T-1)){
    p[t] ~ dunif(0,1)
    #Recapture matrix
    omega[1,1,t] <- 1 - p[t]    # Pr(alive t -> non-detected t)
    omega[1,2,t] <- p[t]       # Pr(alive t -> detected t)
    omega[2,1,t] <- 1          # Pr(dead t -> non-detected t)
    omega[2,2,t] <- 0          # Pr(dead t -> detected t)
  }

  # Likelihood
  for (i in 1:N){
    z[i,first[i]] ~ dcat(delta[1:2])
    for (j in (first[i]+1):T){
      z[i,j] ~ dcat(gamma[z[i,j-1], 1:2, i])
      y[i,j] ~ dcat(omega[z[i,j], 1:2, j-1])
    }
  }
})
```

```

#' Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#' A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)
my.constants

#' Now the data in a list.
my.data <- list(y = y + 1)

#' Specify initial values.
zinit <- y + 1 # non-detection -> alive
zinit[zinit == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                   p = runif(my.constants$T-1,0,1),
                                   z = zinit)
initial.values()

#' Specify the parameters we wish to monitor.
parameters.to.save <- c("beta", "phi_male", "phi_female", "p")
parameters.to.save

#' MCMC details.
n.iter <- 2500
n.burnin <- 1000
n.chains <- 2

#' At last, let's run nimble.
mcmc.phispt <- nimbleMCMC(code = hmm.phispt,
                        constants = my.constants,
                        data = my.data,
                        inits = initial.values,
                        monitors = parameters.to.save,
                        niter = n.iter,
                        nburnin = n.burnin,
                        nchains = n.chains)

#' Examine the results.
MCMCsummary(mcmc.phispt, round = 2)
MCMCtrace(mcmc.phispt, pdf=F)

```

$\phi(\text{sex})$ $p(\text{sex}+t)$

```
##### phi(s)p(s+t)-----
hmm.phisps_t <- nimbleCode({
  # Transition matrix
  for(i in 1:N){
    logit(phi[i])<- beta[sex[i]]
    #Transition matrix
    gamma[1,1,i] <- phi[i]      # Pr(alive t -> alive t+1)
    gamma[1,2,i] <- 1 - phi[i]  # Pr(alive t -> dead t+1)
    gamma[2,1,i] <- 0           # Pr(dead t -> alive t+1)
    gamma[2,2,i] <- 1           # Pr(dead t -> dead t+1)
  }
  # Priors for b1
  beta[1] ~ dnorm(mean = 0, sd = 1.5)
  beta[2] ~ dnorm(mean = 0, sd = 1.5)

  # ilogit for phi
  phi_male <- ilogit(beta[1])
  phi_female <- ilogit(beta[2])

  # Initial state prob.
  delta[1] <- 1      # Pr(alive t = 1) = 1
  delta[2] <- 0      # Pr(dead t = 1) = 0

  ## Observation matrix
  for(i in 1:N){
    for(t in 1:(T-1)){
      logit(p[i,t]) <- beta2[sex[i]] + lambda[t]
      #Observation matrix
      omega[1,1,i,t] <- 1 - p[i,t]    # Pr(alive t -> non-detected t)
      omega[1,2,i,t] <- p[i,t]        # Pr(alive t -> detected t)
      omega[2,1,i,t] <- 1              # Pr(dead t -> non-detected t)
      omega[2,2,i,t] <- 0              # Pr(dead t -> detected t)
    }
  }

  # Priors for b2
  beta2[1] ~ dnorm(mean = 0, sd = 1.5)
  beta2[2] ~ dnorm(mean = 0, sd = 1.5)

  # Time fixed effect
  for(t in 1:(T-1)){
    lambda[t] ~ dnorm(0, 1.5)
  }

  # Recapture probability.
  for(t in 1:(T-1)){
    p_male[t] <- ilogit(beta2[1] + lambda[t])
    p_female[t] <- ilogit(beta2[2] + lambda[t])
  }

  ## Likelihood
```

```

for (i in 1:N){
  z[i,first[i]] ~ dcat(delta[1:2])
  for (j in (first[i]+1):T){
    z[i,j] ~ dcat(gamma[z[i,j-1], 1:2, i])
    y[i,j] ~ dcat(omega[z[i,j], 1:2, i, j-1])
  }
}
})

#' Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#' A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)
my.constants

#' Now the data in a list.
my.data <- list(y = y + 1)

#' Specify initial values.
zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                   beta2 = rnorm(2,0,1),
                                   lambda = rnorm(6, 0, 1),
                                   z = zinits)
initial.values()

#' Specify the parameters we wish to monitor.
parameters.to.save <- c("beta", "lambda", "phi_male", "phi_female", "p_male", "p_female")
parameters.to.save

#' MCMC details.
n.iter <- 2500
n.burnin <- 1000
n.chains <- 2

#' At last, let's run nimble.
mcmc.phisps_t <- nimbleMCMC(code = hmm.phisps_t,
                            constants = my.constants,
                            data = my.data,
                            inits = initial.values,
                            monitors = parameters.to.save,
                            niter = n.iter,

```

```

                                nburnin = n.burnin,
                                nchains = n.chains)

#' Examine the results.
MCMCsummary(mcmc.phisps_t, round = 2)
MCMCtrace(mcmc.phisps_t, params = "p", pdf=F)

```

$\phi(\text{sex}) p(\text{sex} \cdot t)$

```

##### phi(s)p(s*t)-----
hmm.phispst <- nimbleCode({

  # Initial state prob.
  delta[1] <- 1          # Pr(alive t = 1) = 1
  delta[2] <- 0          # Pr(dead t = 1) = 0

  #Survival
  for(i in 1:N){
    logit(phi[i])<- beta[sex[i]]
    #Survival matrix
    gamma[1,1,i] <- phi[i]      # Pr(alive t -> alive t+1)
    gamma[1,2,i] <- 1 - phi[i]  # Pr(alive t -> dead t+1)
    gamma[2,1,i] <- 0           # Pr(dead t -> alive t+1)
    gamma[2,2,i] <- 1           # Pr(dead t -> dead t+1)
  }

  # Priors for b1
  beta[1] ~ dnorm(mean = 0, sd = 1.5)
  beta[2] ~ dnorm(mean = 0, sd = 1.5)

  # ilogit for phi
  phi_male <- ilogit(beta[1])
  phi_female <- ilogit(beta[2])

  #Recapture
  for(i in 1:N){
    for(t in 1:(T-1)){
      logit(p[i,t]) <- beta2[sex[i]] + lambda[t] + kappa[sex[i],t]
      #Recapture matrix
      omega[1,1,i,t] <- 1 - p[i,t]  # Pr(alive t -> non-detected t)
      omega[1,2,i,t] <- p[i,t]      # Pr(alive t -> detected t)
      omega[2,1,i,t] <- 1           # Pr(dead t -> non-detected t)
      omega[2,2,i,t] <- 0           # Pr(dead t -> detected t)
    }
  }

  # Priors for b2
  beta2[1] ~ dnorm(mean = 0, sd = 1.5)
  beta2[2] ~ dnorm(mean = 0, sd = 1.5)

  # Time fixed effect
  for(t in 1:(T-1)){

```

```

    lambda[t] ~ dnorm(0, 1.5)
  }

  # Time as random effect for the interaction
  t.sigma ~ dunif(0, 10)
  for(i in 1:2){
    for(t in 1:(T-1)){
      kappa[i,t] ~ dnorm(0, sd = t.sigma)
    }
  }

  # Recapture probability.
  for(t in 1:(T-1)){
    p_male[t] <- ilogit(beta2[1] + lambda[t] + kappa[1,t])
    p_female[t] <- ilogit(beta2[2] + lambda[t] + kappa[2,t])
  }

  ## Likelihood
  for (i in 1:N){
    z[i,first[i]] ~ dcat(delta[1:2])
    for (j in (first[i]+1):T){
      z[i,j] ~ dcat(gamma[z[i,j-1], 1:2, i])
      y[i,j] ~ dcat(omega[z[i,j], 1:2, i, j-1])
    }
  }
})

#' Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#' A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)
my.constants

#' Now the data in a list.
my.data <- list(y = y + 1)

#' Specify initial values.
zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                  beta2 = rnorm(2,0,1),
                                  lambda = rnorm(6, 0, 1)),
                                  t.sigma = runif(1,0,1),
                                  kappa = matrix(rnorm(12, 0, 1), 2, 6),
                                  z = zinits)

```

```

initial.values()

#' Specify the parameters we wish to monitor.
parameters.to.save <- c("beta", "lambda", "beta2", "phi_male",
                        "phi_female", "p_male", "p_female")
parameters.to.save

#' MCMC details.
n.iter <- 10000
n.burnin <- 1000
n.chains <- 2

#' At last, let's run nimble.
mcmc.phispst <- nimbleMCMC(code = hmm.phispst,
                           constants = my.constants,
                           data = my.data,
                           inits = initial.values,
                           monitors = parameters.to.save,
                           niter = n.iter,
                           nburnin = n.burnin,
                           nchains = n.chains)

#' Examine the results.
MCMCsummary(mcmc.phispst, round = 2)
MCMCtrace(mcmc.phispst, params = "p", pdf=F)

```

Survival with interaction of sex and time $\sim \phi(\text{sex} \cdot \text{time})$

$\phi(\text{sex} \cdot \text{time})$ $p(\cdot)$

```

##### phi(s*t)p(.)-----
hmm.phistps <- nimbleCode({

  #Initial state prob
  delta[1] <- 1          # Pr(alive t = 1) = 1
  delta[2] <- 0          # Pr(dead t = 1) = 0

  #Survival
  for(i in 1:N){
    for(t in 1:(T-1)){
      logit(phi[i,t]) <- beta[sex[i]] + lambda[t] + kappa[sex[i],t]
      #Survival matrix
      gamma[1,1,i,t] <- phi[i,t]          # Pr(alive t -> alive t+1)
      gamma[1,2,i,t] <- 1 - phi[i,t]      # Pr(alive t -> dead t+1)
      gamma[2,1,i,t] <- 0                  # Pr(dead t -> alive t+1)
      gamma[2,2,i,t] <- 1                  # Pr(dead t -> dead t+1)
    }
  }
}

```

```

#Recapture
p ~ dunif(0, 1) # prior recapture
#Recapture matrix
omega[1,1] <- 1 - p # Pr(alive t -> non-detected t)
omega[1,2] <- p # Pr(alive t -> detected t)
omega[2,1] <- 1 # Pr(dead t -> non-detected t)
omega[2,2] <- 0 # Pr(dead t -> detected t)

## Priors for b1
beta[1] ~ dnorm(mean = 0, sd = 1.5)
beta[2] ~ dnorm(mean = 0, sd = 1.5)

#Time fixed effect
for(t in 1:(T-1)){
  lambda[t] ~ dnorm(mean = 0, sd = 1.5)
}

# Time as random for the interaction
t.sigma ~ dunif(0, 10)
for(i in 1:2){
  for(t in 1:(T-1)){
    kappa[i,t] ~ dnorm(mean = 0, sd = t.sigma)
  }
}

# ilogit for phi
for (t in 1:(T-1)){
  phi_male[t] <- ilogit(beta[1]+ lambda[t] + kappa[1,t])
  phi_female[t] <- ilogit(beta[2] + lambda[t] + kappa[2,t])
}

# Likelihood
for (i in 1:N){
  z[i,first[i]] ~ dcat(delta[1:2])
  for (j in (first[i]+1):T){
    z[i,j] ~ dcat(gamma[z[i,j-1], 1:2,i, j-1])
    y[i,j] ~ dcat(omega[z[i,j], 1:2])
  }
}
})

#' Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#' A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)
my.constants

```



```

# Now the data in a list.
my.data <- list(y = y + 1)

# Specify initial values.
zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                   p = runif(1,0,1),
                                   lambda = rnorm(6, 0, 1),
                                   t.sigma = runif(1,0,1),
                                   kappa = matrix(rnorm(12, 0, 1), 2, 6),
                                   z = zinits)

initial.values()

# Specify the parameters we wish to monitor.
parameters.to.save <- c("phi_male", "phi_female", "p")
parameters.to.save

# MCMC details.
n.iter <- 10000
n.burnin <- 1000
n.chains <- 2

# At last, let's run nimble.
mcmc.hmm.phistps <- nimbleMCMC(code = hmm.phistps,
                                constants = my.constants,
                                data = my.data,
                                inits = initial.values,
                                monitors = parameters.to.save,
                                niter = n.iter,
                                nburnin = n.burnin,
                                nchains = n.chains)

# Examine the results.
MCMCsummary(mcmc.hmm.phistps, round = 2)
MCMCtrace(mcmc.hmm.phistps, params = "all", pdf=F)

```

$\phi(\text{sex} \cdot \text{time}) p(\text{sex})$

```

##### phi(s*t)p(s)-----
hmm.phistps <- nimbleCode({

  #Initial state prob
  delta[1] <- 1      # Pr(alive t = 1) = 1
  delta[2] <- 0      # Pr(dead t = 1) = 0

  #Survival

```

```

for(i in 1:N){
  for(t in 1:(T-1)){
    logit(phi[i,t]) <- beta[sex[i]] + lambda[t] + kappa[sex[i],t]
    #Survival matrix
    gamma[1,1,i,t] <- phi[i,t]      # Pr(alive t -> alive t+1)
    gamma[1,2,i,t] <- 1 - phi[i,t]  # Pr(alive t -> dead t+1)
    gamma[2,1,i,t] <- 0              # Pr(dead t -> alive t+1)
    gamma[2,2,i,t] <- 1              # Pr(dead t -> dead t+1)
  }
}

# Priors for b1
beta[1] ~ dnorm(mean = 0, sd = 1.5)
beta[2] ~ dnorm(mean = 0, sd = 1.5)

#Recapture
for(i in 1:N){
  for(t in 1:(T-1)){
    logit(p[i]) <- beta2[sex[i]]
    #Recapture matrix
    omega[1,1,i] <- 1 - p[i]      # Pr(alive t -> non-detected t)
    omega[1,2,i] <- p[i]          # Pr(alive t -> detected t)
    omega[2,1,i] <- 1             # Pr(dead t -> non-detected t)
    omega[2,2,i] <- 0             # Pr(dead t -> detected t)
  }
}

## Priors for b1 and b2
beta[1] ~ dnorm(mean = 0, sd = 1.5)
beta[2] ~ dnorm(mean = 0, sd = 1.5)
beta2[1] ~ dnorm(mean = 0, sd = 1.5)
beta2[2] ~ dnorm(mean = 0, sd = 1.5)

#Time fixed effect
for(t in 1:(T-1)){
  lambda[t] ~ dnorm(mean = 0, sd = 1.5)
}

# Time as random for the interaction
t.sigma ~ dunif(0, 10)
for(i in 1:2){
  for(t in 1:(T-1)){
    kappa[i,t] ~ dnorm(mean = 0, sd = t.sigma)
  }
}

# ilogit for phi
for (t in 1:(T-1)){
  phi_male[t] <- ilogit(beta[1]+ lambda[t] + kappa[1,t])
  phi_female[t] <- ilogit(beta[2] + lambda[t] + kappa[2,t])
}

#ilogit for p
p_male <- ilogit(beta2[1])

```

```

p_female <- ilogit(beta2[2])

#Likelihood
for (i in 1:N){
  z[i,first[i]] ~ dcat(delta[1:2])
  for (j in (first[i]+1):T){
    z[i,j] ~ dcat(gamma[z[i,j-1], 1:2,i, j-1])
    y[i,j] ~ dcat(omega[z[i,j], 1:2, i])
  }
}
})

#' Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#' A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)
my.constants

#' Now the data in a list.
my.data <- list(y = y + 1)

#' Specify initial values.
zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                  beta2 = rnorm(2,0,1),
                                  lambda = rnorm(6, 0, 1),
                                  t.sigma = runif(1,0,1),
                                  kappa = matrix(rnorm(12, 0, 1), 2, 6),
                                  z = zinits)
initial.values()

#' Specify the parameters we wish to monitor.
parameters.to.save <- c("phi_male", "phi_female", "p")
parameters.to.save

#' MCMC details.
n.iter <- 10000
n.burnin <- 1000
n.chains <- 2

#' At last, let's run nimble.
mcmc.hmm.phistps <- nimbleMCMC(code = hmm.phistps,
                              constants = my.constants,

```

```

data = my.data,
inits = initial.values,
monitors = parameters.to.save,
niter = n.iter,
nburnin = n.burnin,
nchains = n.chains)

#' Examine the results.
MCMCsummary(mcmc.hmm.phistps, round = 2)
MCMCtrace(mcmc.hmm.phistps, params = "all", pdf=F)

```

$\phi(\text{sex} \cdot \text{time}) p(t)$

```

##### phi(s*t)p(t)-----
hmm.phistpt <- nimbleCode({

  #Initial state prob
  delta[1] <- 1      # Pr(alive t = 1) = 1
  delta[2] <- 0      # Pr(dead t = 1) = 0

  #Survival
  for(i in 1:N){
    for(t in 1:(T-1)){
      logit(phi[i,t]) <- beta[sex[i]] + lambda[t] + kappa[sex[i],t]
      #Survival matrix
      gamma[1,1,i,t] <- phi[i,t]      # Pr(alive t -> alive t+1)
      gamma[1,2,i,t] <- 1 - phi[i,t]  # Pr(alive t -> dead t+1)
      gamma[2,1,i,t] <- 0              # Pr(dead t -> alive t+1)
      gamma[2,2,i,t] <- 1              # Pr(dead t -> dead t+1)
    }
  }

  #Recapture
  for(t in 1:(T-1)){
    p[t] ~ dunif(0, 1) # prior for p
    #Recapture matrix
    omega[1,1,i] <- 1 - p[i]      # Pr(alive t -> non-detected t)
    omega[1,2,i] <- p[i]         # Pr(alive t -> detected t)
    omega[2,1,i] <- 1             # Pr(dead t -> non-detected t)
    omega[2,2,i] <- 0             # Pr(dead t -> detected t)
  }

  ## Priors for b1
  beta[1] ~ dnorm(mean = 0, sd = 1.5)
  beta[2] ~ dnorm(mean = 0, sd = 1.5)

  #Time fixed effect
  for(t in 1:(T-1)){
    lambda[t] ~ dnorm(mean = 0, sd = 1.5)
  }

  # Time as random for the interaction

```

```

t.sigma ~ dunif(0, 10)
for(i in 1:2){
  for(t in 1:(T-1)){
    kappa[i,t] ~ dnorm(mean = 0, sd = t.sigma)
  }
}

# ilogit for phi
for (t in 1:(T-1)){
  phi_male[t] <- ilogit(beta[1]+ lambda[t] + kappa[1,t])
  phi_female[t] <- ilogit(beta[2] + lambda[t] + kappa[2,t])
}

#Likelihood
for (i in 1:N){
  z[i,first[i]] ~ dcat(delta[1:2])
  for (j in (first[i]+1):T){
    z[i,j] ~ dcat(gamma[z[i,j-1], 1:2,i, j-1])
    y[i,j] ~ dcat(omega[z[i,j], 1:2, i])
  }
}
})

#' Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#' A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)
my.constants

#' Now the data in a list.
my.data <- list(y = y + 1)

#' Specify initial values.
zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                  p = runif(my.constants$T-1, 0, 1),
                                  lambda = rnorm(6, 0, 1),
                                  t.sigma = runif(1,0,1),
                                  kappa = matrix(rnorm(12, 0, 1), 2, 6),
                                  z = zinits)
initial.values()

#' Specify the parameters we wish to monitor.
parameters.to.save <- c("phi_male", "phi_female", "p")

```

```

parameters.to.save

#' MCMC details.
n.iter <- 10000
n.burnin <- 1000
n.chains <- 2

#' At last, let's run nimble.
mcmc.hmm.phistpt <- nimbleMCMC(code = hmm.phistpt,
                                constants = my.constants,
                                data = my.data,
                                inits = initial.values,
                                monitors = parameters.to.save,
                                niter = n.iter,
                                nburnin = n.burnin,
                                nchains = n.chains)

#' Examine the results.
MCMCsummary(mcmc.hmm.phistpt, round = 2)
MCMCtrace(mcmc.hmm.phistpt, params = "all", pdf=F)

```

$\phi(\text{sex} \cdot \text{time})$ $p(\text{sex} + \text{time})$

```

##### phi(s*t)p(s+t)-----
hmm.phistpst <- nimbleCode({
  for(i in 1:N){
    for(t in 1:(T-1)){
      logit(phi[i,t]) <- beta[sex[i]] + lambda[t] + kappa[sex[i],t]
      gamma[1,1,i,t] <- phi[i,t]      # Pr(alive t -> alive t+1)
      gamma[1,2,i,t] <- 1 - phi[i,t]  # Pr(alive t -> dead t+1)
      gamma[2,1,i,t] <- 0              # Pr(dead t -> alive t+1)
      gamma[2,2,i,t] <- 1              # Pr(dead t -> dead t+1)
    }
  }

  #Initial state prob
  delta[1] <- 1      # Pr(alive t = 1) = 1
  delta[2] <- 0      # Pr(dead t = 1) = 0

  for(i in 1:N){
    for(t in 1:(T-1)){
      logit(p[i,t]) <- beta2[sex[i]] + lambda2[t]
      omega[1,1,i,t] <- 1 - p[i,t]    # Pr(alive t -> non-detected t)
      omega[1,2,i,t] <- p[i,t]        # Pr(alive t -> detected t)
      omega[2,1,i,t] <- 1              # Pr(dead t -> non-detected t)
      omega[2,2,i,t] <- 0              # Pr(dead t -> detected t)
    }
  }

  ## Priors for b1 b2 b3 and b4
  beta[1] ~ dnorm(mean = 0, sd = 1.5)

```

```

beta[2] ~ dnorm(mean = 0, sd = 1.5)
beta2[1] ~ dnorm(mean = 0, sd = 1.5)
beta2[2] ~ dnorm(mean = 0, sd = 1.5)

#Time fixed effect
for(t in 1:(T-1)){
  lambda[t] ~ dnorm(mean = 0, sd = 1.5)
  lambda2[t] ~ dnorm(mean = 0, sd = 1.5)
}

# Time as random for the interaction
t.sigma1 ~ dunif(0, 10)

for(i in 1:2){
  for(t in 1:(T-1)){
    kappa[i,t] ~ dnorm(mean = 0, sd = t.sigma1)
  }
}

# ilogit for phi and p
for (t in 1:(T-1)){
  phi_male[t] <- ilogit(beta[1]+ lambda[t] + kappa[1,t])
  phi_female[t] <- ilogit(beta[2] + lambda[t] + kappa[2,t])
  p_male[t] <- ilogit(beta2[1] + lambda2[t])
  p_female[t] <- ilogit(beta2[2] + lambda2[t])
}

# Pr(dead t -> detected t)
for (i in 1:N){
  z[i,first[i]] ~ dcat(delta[1:2])
  for (j in (first[i]+1):T){
    z[i,j] ~ dcat(gamma[z[i,j-1], 1:2,i, j-1])
    y[i,j] ~ dcat(omega[z[i,j], 1:2, i, j-1])
  }
}
})

#' Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#' A list with constants.
my.constants <- list(N = nrow(y),
                     T = ncol(y),
                     first = first,
                     sex = sex)
my.constants

#' Now the data in a list.
my.data <- list(y = y + 1)

```

```

# Specify initial values.
zinit1 <- y + 1 # non-detection -> alive
zinit2[zinit2 == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                beta2 = rnorm(2,0,1),
                                lambda = rnorm(6, 0, 1),
                                lambda2 = rnorm(6, 0, 1),
                                t.sigma1 = runif(1,0,1),
                                kappa = matrix(rnorm(12, 0, 1), 2, 6),
                                z = zinit1)

initial.values()

# Specify the parameters we wish to monitor.
parameters.to.save <- c("phi_male", "phi_female", "p_male", "p_female")
parameters.to.save

# MCMC details.
n.iter <- 10000
n.burnin <- 1000
n.chains <- 2

# At last, let's run nimble.
mcmc.phistpst <- nimbleMCMC(code = hmm.phistpst,
                           constants = my.constants,
                           data = my.data,
                           inits = initial.values,
                           monitors = parameters.to.save,
                           niter = n.iter,
                           nburnin = n.burnin,
                           nchains = n.chains)

```

$\phi(\text{sex} \cdot \text{time})$ $p(\text{sex} \cdot \text{time})$

```

## PHI(s*t) -----
#### phi(s*t)p(s*t)-----
hmm.phistpst <- nimbleCode({

  #Initial state prob
  delta[1] <- 1          # Pr(alive t = 1) = 1
  delta[2] <- 0          # Pr(dead t = 1) = 0

  #Survival
  for(i in 1:N){
    for(t in 1:(T-1)){
      logit(phi[i,t]) <- beta[sex[i]] + lambda[t] + kappa[sex[i],t]
      #Survival matrix
      gamma[1,1,i,t] <- phi[i,t]          # Pr(alive t -> alive t+1)
      gamma[1,2,i,t] <- 1 - phi[i,t]      # Pr(alive t -> dead t+1)
      gamma[2,1,i,t] <- 0                  # Pr(dead t -> alive t+1)
    }
  }
})

```



```

    gamma[2,2,i,t] <- 1          # Pr(dead t -> dead t+1)
  }
}

#Recapture
for(i in 1:N){
  for(t in 1:(T-1)){
    logit(p[i,t]) <- beta2[sex[i]] + lambda2[t] + kappa2[sex[i],t]
    #Recapture matrix
    omega[1,1,i,t] <- 1 - p[i,t]      # Pr(alive t -> non-detected t)
    omega[1,2,i,t] <- p[i,t]         # Pr(alive t -> detected t)
    omega[2,1,i,t] <- 1              # Pr(dead t -> non-detected t)
    omega[2,2,i,t] <- 0              # Pr(dead t -> detected t)
  }
}

## Priors for betas
beta[1] ~ dnorm(mean = 0, sd = 1.5)
beta[2] ~ dnorm(mean = 0, sd = 1.5)
beta2[1] ~ dnorm(mean = 0, sd = 1.5)
beta2[2] ~ dnorm(mean = 0, sd = 1.5)

#Time fixed effect
for(t in 1:(T-1)){
  lambda[t] ~ dnorm(mean = 0, sd = 1.5)
  lambda2[t] ~ dnorm(mean = 0, sd = 1.5)
}

# Time as random for the interaction
t.sigma1 ~ dunif(0, 10)
t.sigma2 ~ dunif(0, 10)

for(i in 1:2){
  for(t in 1:(T-1)){
    kappa[i,t] ~ dnorm(mean = 0, sd = t.sigma1)
    kappa2[i,t] ~ dnorm(mean = 0, sd = t.sigma2)
  }
}

# ilogit for phi and p
for (t in 1:(T-1)){
  phi_male[t] <- ilogit(beta[1]+ lambda[t] + kappa[1,t])
  phi_female[t] <- ilogit(beta[2] + lambda[t] + kappa[2,t])
  p_male[t] <- ilogit(beta2[1] + lambda2[t] + kappa2[2,t])
  p_female[t] <- ilogit(beta2[2] + lambda2[t] + kappa2[2,t])
}

# Likelihood
for (i in 1:N){
  z[i,first[i]] ~ dcat(delta[1:2])
  for (j in (first[i]+1):T){
    z[i,j] ~ dcat(gamma[z[i,j-1], 1:2,i, j-1])
    y[i,j] ~ dcat(omega[z[i,j], 1:2, i, j-1])
  }
}

```

```

    }
  }
})

#' Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#' A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)
my.constants

#' Now the data in a list.
my.data <- list(y = y + 1)

#' Specify initial values.
zinit <- y + 1 # non-detection -> alive
zinit[zinit == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                  beta2 = rnorm(2,0,1),
                                  lambda = rnorm(6, 0, 1),
                                  lambda2 = rnorm(6, 0, 1),
                                  t.sigma1 = runif(1,0,1),
                                  t.sigma2 = runif(1,0,1),
                                  kappa = matrix(rnorm(12, 0, 1), 2, 6),
                                  kappa2 = matrix(rnorm(12, 0, 1), 2, 6),
                                  z = zinit)
initial.values()

#' Specify the parameters we wish to monitor.
parameters.to.save <- c("phi_male", "phi_female", "p_male", "p_female")
parameters.to.save

#' MCMC details.
n.iter <- 10000
n.burnin <- 1000
n.chains <- 2

#' At last, let's run nimble.
mcmc.phistpst <- nimbleMCMC(code = hmm.phistpst,
                           constants = my.constants,
                           data = my.data,
                           inits = initial.values,
                           monitors = parameters.to.save,
                           niter = n.iter,

```

```

                                nburnin = n.burnin,
                                nchains = n.chains)

#' Examine the results.
MCMCsummary(mcmc.phistpst, round = 2)
MCMCtrace(mcmc.phistpst, params = "all", pdf=F)
.....

### Additive effect of sex and time in survival and recapture
###  $\phi(\text{sex}+\text{time})$   $p(\text{sex}+\text{time})$ 

## PHI(s+t) -----
#####  $\phi(s+t)p(s+t)$ -----
hmm.phistpst <- nimbleCode({

  #Initial state prob
  delta[1] <- 1          # Pr(alive t = 1) = 1
  delta[2] <- 0          # Pr(dead t = 1) = 0

  #Survival
  for(i in 1:N){
    for(t in 1:(T-1)){
      logit(phi[i,t]) <- beta[sex[i]] + lambda[t]
      #Survival matrix
      gamma[1,1,i,t] <- phi[i,t]      # Pr(alive t -> alive t+1)
      gamma[1,2,i,t] <- 1 - phi[i,t]  # Pr(alive t -> dead t+1)
      gamma[2,1,i,t] <- 0              # Pr(dead t -> alive t+1)
      gamma[2,2,i,t] <- 1              # Pr(dead t -> dead t+1)
    }
  }

  #Recapture
  for(i in 1:N){
    for(t in 1:(T-1)){
      logit(p[i,t]) <- beta2[sex[i]] + lambda2[t]
      #Recapture matrix
      omega[1,1,i,t] <- 1 - p[i,t]    # Pr(alive t -> non-detected t)
      omega[1,2,i,t] <- p[i,t]        # Pr(alive t -> detected t)
      omega[2,1,i,t] <- 1              # Pr(dead t -> non-detected t)
      omega[2,2,i,t] <- 0              # Pr(dead t -> detected t)
    }
  }

  ## Priors for b1 b2
  beta[1] ~ dnorm(mean = 0, sd = 1.5)
  beta[2] ~ dnorm(mean = 0, sd = 1.5)
  beta2[1] ~ dnorm(mean = 0, sd = 1.5)
  beta2[2] ~ dnorm(mean = 0, sd = 1.5)

  #Time fixed effect
  for (t in 1:(T-1)){
    lambda[t] ~ dnorm(mean = 0, sd = 1.5)
    lambda2[t] ~ dnorm(mean = 0, sd = 1.5)
  }
}

```

```

#ilogit for phi and p
for(t in 1:(T-1)){
  phi_male[t] <- ilogit(beta[1]+ lambda[t])
  phi_female[t] <- ilogit(beta[2] + lambda[t])
  p_male[t] <- ilogit(beta2[1] + lambda2[t])
  p_female[t] <- ilogit(beta2[2] + lambda2[t])
}

#Likelihood
for (i in 1:N){
  z[i,first[i]] ~ dcat(delta[1:2])
  for (j in (first[i]+1):T){
    z[i,j] ~ dcat(gamma[z[i,j-1], 1:2, i, j-1])
    y[i,j] ~ dcat(omega[z[i,j], 1:2, i, j-1])
  }
}
})

#' Get the occasion of first capture for all individuals.
first <- apply(y, 1, function(x) min(which(x !=0)))
first

#' A list with constants.
my.constants <- list(N = nrow(y),
                    T = ncol(y),
                    first = first,
                    sex = sex)
my.constants

#' Now the data in a list.
my.data <- list(y = y + 1)

#' Specify initial values.
zinits <- y + 1 # non-detection -> alive
zinits[zinits == 2] <- 1 # dead -> alive
initial.values <- function() list(beta = rnorm(2,0,1),
                                beta2 = rnorm(2,0,1),
                                lambda = rnorm(6,0,1),
                                lambda2 = rnorm(6,0,1),
                                z = zinits)
initial.values()

#' Specify the parameters we wish to monitor.
parameters.to.save <- c("phi_male", "phi_female", "p_male", "p_female")
parameters.to.save

#' MCMC details.
n.iter <- 10000
n.burnin <- 1000

```

```

n.chains <- 2

#' At last, let's run nimble.
mcmc.phistpst <- nimbleMCMC(code = hmm.phistpst,
                           constants = my.constants,
                           data = my.data,
                           inits = initial.values,
                           monitors = parameters.to.save,
                           niter = n.iter,
                           nburnin = n.burnin,
                           nchains = n.chains)

#' Examine the results.
MCMCsummary(mcmc.phistpst, round = 2)
MCMCtrace(mcmc.phistpst, params = "all", pdf=F)

```

Reference and acknowledgements

- We would like to thank Oliver Gimenez for giving us total permission to share this code.
- de Valpine P, Turek D, Paciorek C, Anderson-Bergman C, Temple Lang D, Bodik R (2017). Programming with models: writing statistical algorithms for general model structures with NIMBLE. *Journal of Computational and Graphical Statistics*, 26, 403-413. doi: 10.1080/10618600.2016.1172487