Supporting information of the article: The participatory monitoring of the capercaillie in the French Pyrenees

Clément Calenge et al.,
Office Français de la Biodiversité
Saint Benoist – 78610 Auffargis – France.

Contents

Introduction

This vignette is the supplementary material of the article of Calenge et al. (in prep.). The aim of this paper is to estimate the numbers of capercaillie cocks in the 5 geographic regions of the French Pyrenees mountains, for each two-year period between 2010 and 2019. A companion package named `caperpyogm` contains the data and functions used for this paper, and is required to reproduce the calculations in this document. The present document is also available as a vignette of this package. To install this package, first install the package `devtools` and use the function `install_github` to install `caperpyogm`:

```
## If devtools is not yet installed, type
install.packages("devtools")

## Install the package caperpyogm
devtools::install_github("ClementCalenge/caperpyogm", ref="main")
```

Remark: on Windows, it is required to also install the Rtools (https://cran.r-project.org/bin/windows/Rtools/) on your computer to have a working `devtools` package (see https://www.r-project.org/nosvn/pandoc/devtools.html).

Throughout this vignette, we suppose that the reader is familiar with the model developed in this paper. Nevertheless, we present a brief reminder on this model in this vignette. This model combines three datasets:

1. the results of counts of singing capercaillie cocks carried out every two-year period on leks, which allow to model the time changes in the mean cock number on the different types of leks (KAL = known active leks; KIL = known leks with indeterminate activity status; UL = leks unknown at the time of the definition of the sampling design), and in the different geographic regions;

2. the results of the search for new ULs in grid cells randomly selected in the study area. This dataset was required to estimate the number of ULs on our study area.

3. the results of an experiment carried out to estimate the lek detection probability. This experiment involved both experienced or inexperienced observers searching for leks unknown to them (but known to us) in selected grid cells.

We developed two sub-models for (i) the mean number of males actually present on a lek of a given type (KAL, KIL, UL), during a given period and within a given geographic region, and (ii) the number of ULs in each geographic region.

In this document, we describe this study in detail:

- In section 1, we present in a table all mathematical notations used in the article and in the present vignette.

- In section 2, we give a brief reminder on the structure of the model used to estimate the number of capercaillie cocks at different spatial scales, and we show how to use the datasets and functions of the package to reproduce the calculations of the article. Note that all the functions of the package have a very detailed help page explaining how they should be used. This section contains additional analyses not presented in the paper (residual analysis, analysis of the convergence and the mixing of MCMC chains, sensitivity to outliers, etc.).

- In section 3, we demonstrate how a cross-validation approach was used to select the best detection model for the lek censuses.

- In section 4, we study more in detail the influence of the lek # 281 on the population size estimation, because lek is characterized by a strong residual in our model.

- In section 5, we considered in detail the criticisms of the N-mixture models by [BAR17] and [LIN18]. In particular, we used simulations to assess the effect of both unaccounted heterogeneity in detection probability and accidental double counting on the estimated number of males.

- In section 6, we give additional elements describing the history of the monitoring program. In particular, we describe how the discovery that the number of cocks on ULs was larger than expected a priori affected our monitoring program.

1 Notations

We present, in the table below, the notations used in this paper. We distinguish several types of notations:

- D: data used to fit the model
- P: stochastic parameter estimated with MCMC
- N: other notations

In the table below, we give:

- the notation
- the type of notation
- a description of the variable or parameter

Notation	Type	Description
$y_{i,t,k}$	D	Number of detected males during census k of year t on lek i
$b(t)$	N	Index of the two-year period containing year t
$N_{i,b(t)}$	P	Number of males actually present on lek i during the two year-period $b(t)$
$p_{i,t,k}$	P	Probability of detection of a male present on lek i during census k of year t
α_d	P	Intercept of the cock detection model
β_d	P	Slope of the number of observers in the cock detection model
$o_{i,t,k}$	D	Number of observers participating to census k of year t on lek i
$e_{g(i),t}$	P	Random effect of year t on cock detectability in geographic region g
σ_e	P	Standard deviation of random effects of the years on the cock detectability
$\lambda_{i,b(t)}$	P	Expectation of the Poisson distribution used for the number of males on lek i during period $b(t)$
$\kappa_{g(i),\ell(i),b(t)}$	P	Parameter describing the average number of males during period $b(t)$ on a lek of type $\ell(i)$ in region $g(i)$
$\nu_{u(i)}$	P	Random effect of natural unit $u(i)$
η_i	P	Random effect of lek i
$\epsilon_{i,b(t)}$	P	Overdispersion residual
$\mu_{g(i),\ell(i)}$	P	Mean of the average cock numbers during period $b(t)$ on leks of type $\ell(i)$ in region $g(i)$
σ_κ	P	Standard deviation of the average cock numbers during period $b(t)$ on leks of type $\ell(i)$ in region $g(i)$
σ_ν	P	Standard deviation of the random effects of natural units
σ_η	P	Standard deviation of the random effects of leks
σ_ϵ	P	Standard deviation of the overdispersion residuals
z_q	D	Whether the cell q contains an unknown lek detected during a search carried out within the framework of our program
f_q	D	Whether the cell q was sampled in 2018–2019
s_q	D	Whether the cell q contains an unknown lek detected before the sampling of the cell within the framework of our program
b_q	P	Whether the cell q actually contains an unknown lek
r_q	D	Proportion of the cell q covered by the capercaillie potential presence area
h_q	D	Whether the cell q already contains a known lek
π_q	P	Probability that the cell q contains an unknown lek
$a_{g(q)}$	P	Intercept of the model predicting π_q
$b_{g(q)}$	P	Slope of r_q in the model predicting π_q
d	P	Slope of h_q in the model predicting π_q
β	P	Probability of detection of an unknown lek during the search of a grid cell
α	P	Probability that an unknown lek present in the cell was discovered prior to its sampling
S_c	D	Number of known leks present in grid cell c included in the search sector
D_c	D	Number of known leks present in grid cell c detected by the observer
N_c	D	Number of known leks present in grid cell c
δ	P	Detection probability of a lek during a search, given that the lek is in the search sector
$e(c)$	N	Level of experience of the observers searching the cell c
$\zeta_{e(c)}$	P	Probability that the search sector defined by the observer with experience $e(c)$ includes a lek present in a cell
$S_{g\ell}$	D/P	Number of leks of type ℓ in region g (to be estimated for ULs)
Q	D	Number of grid cells in the sampling frame
$n_{g\ell b}$	P	Number of males on leks of type ℓ during period b in region g
n_b	P	Number of males on all leks of the mountain range during period b

2 Model description and model fit

We describe in this section the two sub-models required for the estimation of the numbers of capercaillie cocks in the five geographical regions of the Pyrenees, for each two-years period between 2010 and 2019.

2.1 Count model

2.1.1 Model description

We first describe the sub-model of the mean number of males detected during the lek censuses carried out in the French Pyrenees mountains between 2010 and 2019.

Let $y_{i,t,k}$ be the number of males counted during census k of year t on lek i. We model these counts with a N-mixture model. We assume that the number of detected animals can be described using a binomial distribution:

$$y_{i,t,k} \sim \mathcal{B}(N_{i,b(t)}, p_{i,t,k}) \tag{1}$$

where $N_{i,b(t)}$ is the actual number of males actually present on lek i during the two-years period $b(t)$ including year t, and $p_{i,t,k}$ is the corresponding detection probability. We suppose the following detection model:

$$\text{logit } p_{i,t,k} = \alpha_d + \beta_d \times o_{i,t,k} + e_{g(i),t} \tag{2}$$

where $o_{i,t,k}$ is the number of observers present on lek i during census k of year t, α_d is the intercept of the model, β_d is the slope, and $e_{g(i),t}$ is a Gaussian random effect characterizing year t in the geographic region $g(i)$ where the lek i is located:

$$e_{g(i),t} \sim \mathcal{N}(0, \sigma_e)$$

We discuss alternative detection models in section 3.

Moreover, we assume that the actual number of males $N_{i,b(t)}$ on lek i during the two-year period $b(t)$ that includes year t can be described using a Poisson distribution:

$$N_{i,b(t)} \sim \mathcal{P}(\lambda_{i,b(t)}) \tag{3}$$

And we suppose the following log-linear model for the expectation of this distribution:

$$\log \lambda_{i,b(t)} = \kappa_{g(i),\ell(i),b(t)} + \nu_{u(i)} \times I(\ell(i) = 2) + \eta_i + \epsilon_{i,b(t)} \tag{4}$$

Where $\kappa_{g(i),\ell(i),b(t)}$ is a random intercept characterizing type $\ell(i)$ of lek i (1 for KALs, 2 for KILs, and 3 for ULs) in the geographic region $g(i)$ that contains lek i, during two-years period $b(t)$ that includes year t; $\nu_{u(i)}$ is a random effect characterizing natural unit $u(i)$ that contains the lek i (note that a random effect characterizing the natural unit is only included for KILs, see the article for an explanation); η_i is a random effect characterizing lek i, and $\epsilon_{i,b(t)}$ is a Gaussian overdispersion residual.

We furthermore assume that the following distributions describe the parameters:

$$\begin{aligned}
\kappa_{g(i),\ell(i),b(t)} &\sim \mathcal{N}(\mu_{g(i),\ell(i)}, \sigma_\kappa) \\
\nu_{u(i)} &\sim \mathcal{N}(0, \sigma_\nu) \\
\eta_i &\sim \mathcal{N}(0, \sigma_\eta^{\ell(i)}) \\
\epsilon_{i,b(t)} &\sim \mathcal{N}(0, \sigma_\epsilon)
\end{aligned}$$

Remark: We defined five geographic regions for the KALs (1 = western foothills, 2 = central foothills, 3 = central high range, 4 = eastern high range, 5 = western high range), but since we had less data for the other types of leks, we defined only three geographic regions for the KILs (foothills, western high range, central and eastern high range) and only one for ULs.

2.1.2 Fit of the count model

We load the library `caperpyogm`, which contains the data and the functions that we used to fit the model:

```
library(caperpyogm)
```

All the datasets of the package are "lazy-loaded", i.e. they are immediately available to the user:

```
head(lekcounts)
```

```
##   lek period nbobs nbmales gr type natun year
## 1  81      1    -1       2  3    1     1    1
## 2 238      1     0       0  4    1     1    1
## 3 324      1     0       7  5    1     1    1
## 4 324      1     3      11  5    1     1    2
## 5   1      1     0       2  1    1     1    1
## 6   1      1     0       1  1    1     1    1
```

This data.frame contains the results of the censuses carried out on all leks in the Pyrenees mountains from 2010 to 2019. For each census, this data.frame contains:

- The lek label, numbered from 1 to 330
- The two-year period during which the census occurred
- The number of observers. We have subtracted 2 to the number of observers, to improve the mixing of the MCMC chains.
- The number of males counted on the lek
- The ID of geographic region containing the lek. Note that in this dataset, the geographic regions are not yet recoded (i.e. there are still 5 regions for KILs and ULs; these regions will be recoded into three and one region respectively by the function `dataCount2jags` below).
- The type of the lek (1 = KAL, 2 = KIL, 3 = UL)
- The label of the natural unit containing the lek
- The year during which the census occurred.

We have programmed the model in JAGS:

```
cat(modelCountDetectBinREY)
```

```
## model {
##
##   ## Priors for top parameters of the state model
##   sigmanu~dgamma(0.01,0.01)
##   sigmaepsilon~dgamma(0.01,0.01)
##   sigmakappa~dgamma(0.01,0.01)
##   sigmaREY~dgamma(0.01,0.01)
##
##   ## Priors for top parameters of the detection model
##   interceptpd~dnorm(0,0.1)
##   pente1pd~dnorm(0,0.1)
##
##
##   ## Priors for other parameters
##   for (i in 1:Ntypes) {
##
```

```
##       sigmaeta[i]~dgamma(0.01,0.01)
##
##       for (j in 1:Ngr) {
##          mukappa[j,i]~dnorm(0,0.1)
##          for (k in 1:Nperiods) {
##             kappa[j,i,k]~dnorm(mukappa[j,i], sigmakappa)
##          }
##       }
##
##   }
##
##   ## Random effect period
##   for (i in 1:Ngr) {
##     for (j in 1:Nyears) {
##         REY[i,j]~dnorm(0, sigmaREY)
##     }
##   }
##
##   ## Random effects natural units
##   for (i in 1:Nnatun) {
##      nu[i]~dnorm(0,sigmanu)
##   }
##
##   ## Expectation of the number of males
##   for (i in 1:Nleks) {
##       eta[i]~dnorm(0,sigmaeta[ellp[i]])
##       for (j in 1:Nperiods) {
##          epsilon[i,j]~dnorm(0,sigmaepsilon)
##          loglambda[i,j] <- kappa[gr[i],ellp[i],j]+nu[natun[i]]*ell2[i]+eta[i]+epsilon[i,j]
##          log(lambda[i,j]) <- loglambda[i,j]
##       }
##    }
##
##
##
##   ## Likelihood
##   for (i in 1:Nlekperiods) {
##     N[i]~dpois(lambda[lek[i], period[i]]  )
##     for (k in 1:repetition[i]) {
##         logit(pd[i,k]) <- interceptpd + pente1pd*observers[i,k] + REY[gr[lek[i]],year[i,k]]
##         y[i,k]~dbin(pd[i,k], N[i])
##     }
##   }
## }
```

We now use the function `dataCount2jags` to prepare the data to fit the JAGS model. Note that this functions recodes the 5 regions into three regions for the KILs (region 1 = foothills ; region 2 = eastern and central high range; region 5 = western high range) and one region for the ULs (region 1) :

```
dataList <- dataCount2jags(lekcounts$lek, lekcounts$period,
                           lekcounts$nbobs, lekcounts$nbmales,
                           lekcounts$gr, as.numeric(factor(lekcounts$type)),
                           lekcounts$natun, lekcounts$year)
```

We can now use the function `fitModelCount` to fit this model. WARNING: THIS CALCULATION TAKES A VERY LONG TIME (several hours) !!!! Note that we have included the results of this

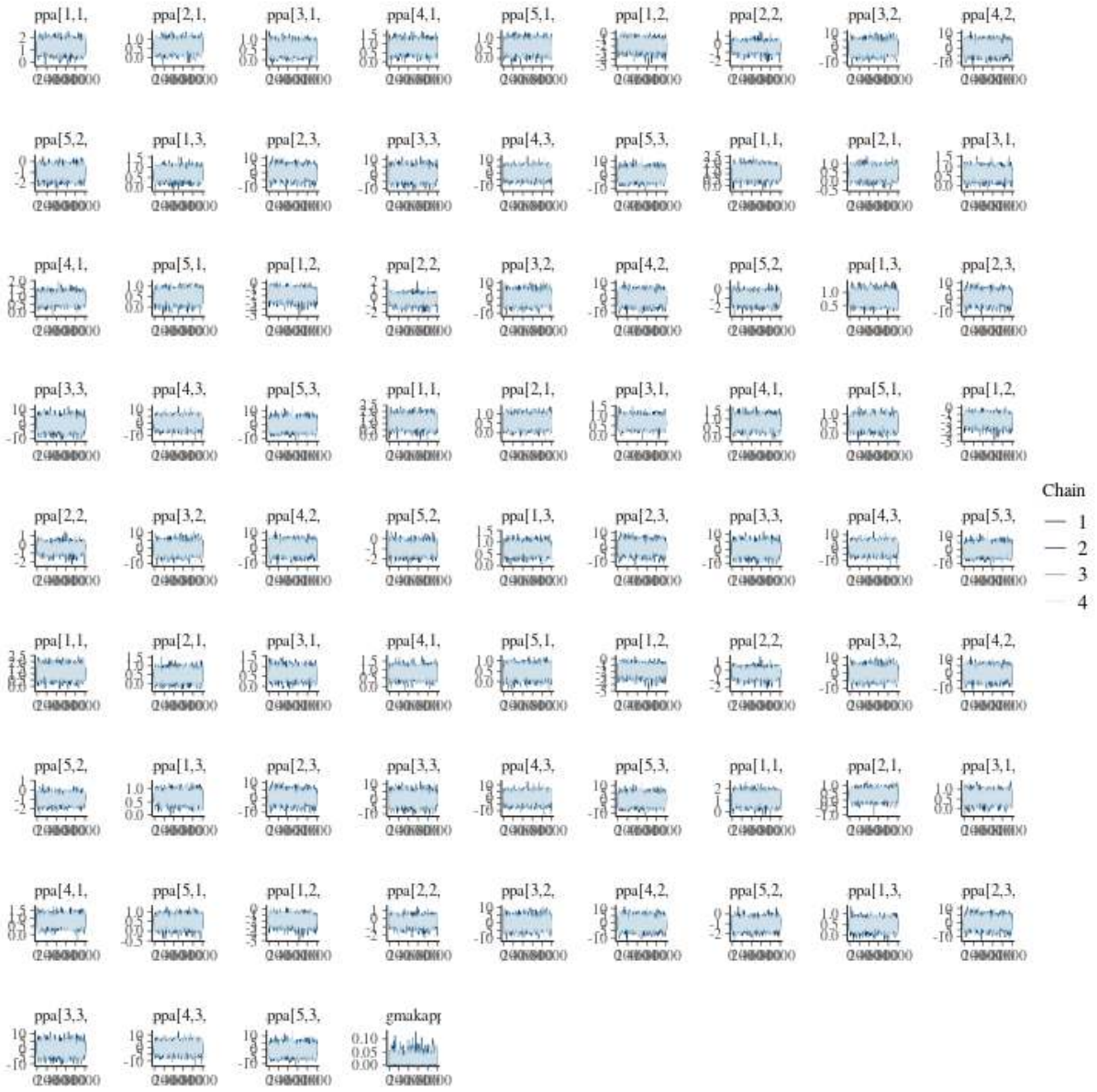
calculation as a dataset of the package, so that the reader does not need to launch this function to reproduce further calculations:

```
coefModelCountDetectBinREY <- fitModelCount(dataList, "modelCountDetectBinREY")
```

2.1.3 MCMC chains mixing

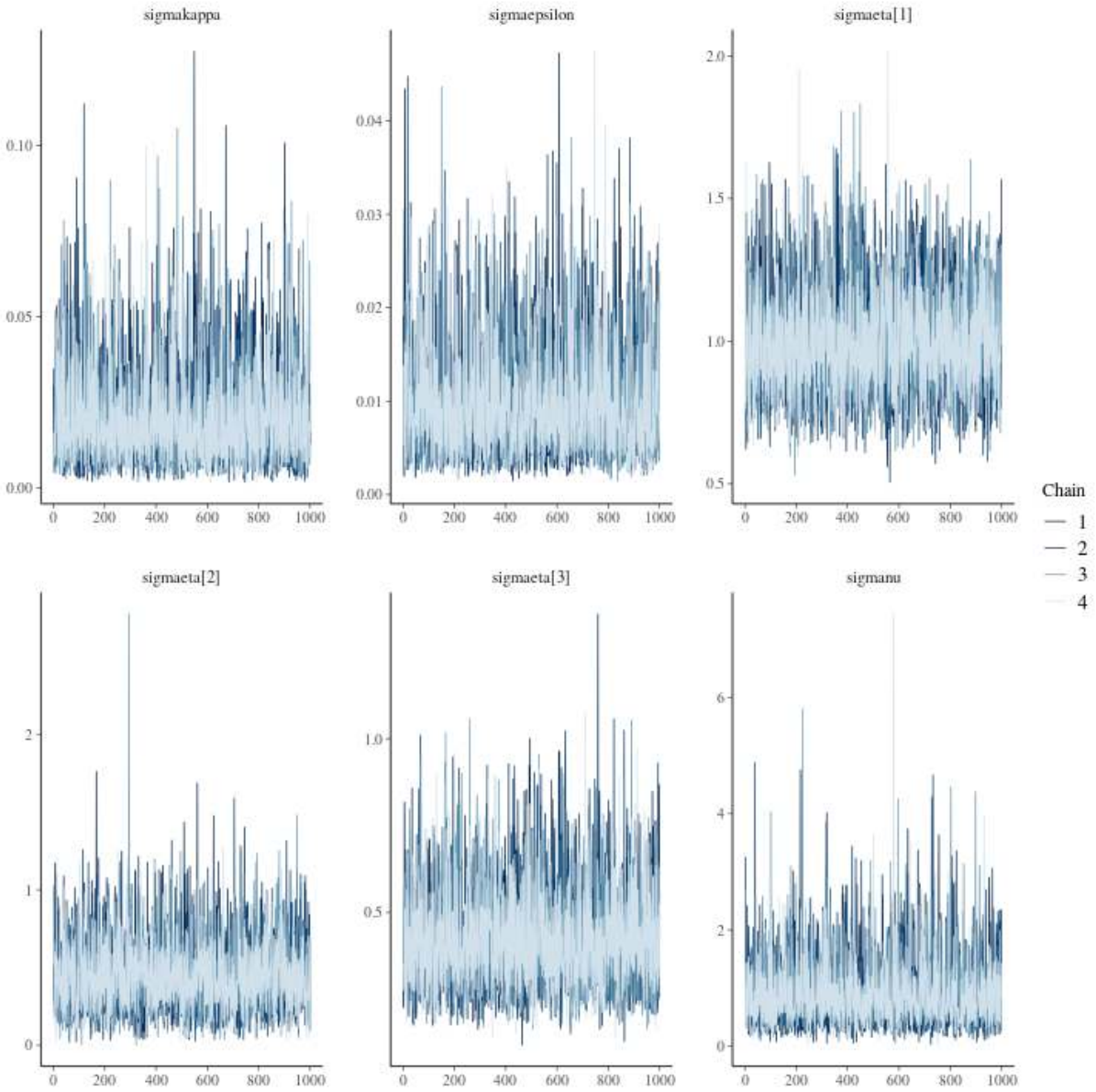
Once the MCMC samples have been obtained, we can plot the chain for a visual examination of the mixing. We present the traceplot for the parameters κ below:

```
library(bayesplot)
library(MCMCvis)
cm <- MCMCchains(coefModelCountDetectBinREY,
                 params=c("kappa","sigmakappa","sigmaepsilon","sigmaeta",
                          "sigmanu", "interceptpd","pente1pd","sigmaREY"),
                 mcmc.list=TRUE)
for (i in 1:length(cm)) {
    for (j in c("sigmakappa","sigmaepsilon","sigmaeta[1]",
                "sigmaeta[2]","sigmaeta[3]","sigmanu","sigmaREY"))
        cm[[i]][,j] <- 1/cm[[i]][,j]
}
mcmc_trace(cm, regex_pars="kappa")
```



We also present the traceplot for the variances of the state model:

```
mcmc_trace(cm, regex_pars=c("sigmakappa","sigmaepsilon","sigmaeta",
                            "sigmanu"))
```



Finally, we present the parameters for the detection model:

```
mcmc_trace(coefModelCountDetectBinREY,
           regex_pars=c("REY","interceptpd","pente1pd","sigmaREY"))
```

We also calculate the statistic of Gelman and Rubin (1992) for the parameters. To save some space, we do not show the results of this diagnostic in this vignette and leave it to the reader to check the correct mixing based on this diagnostic:

```
gelman.diag(cm)
```

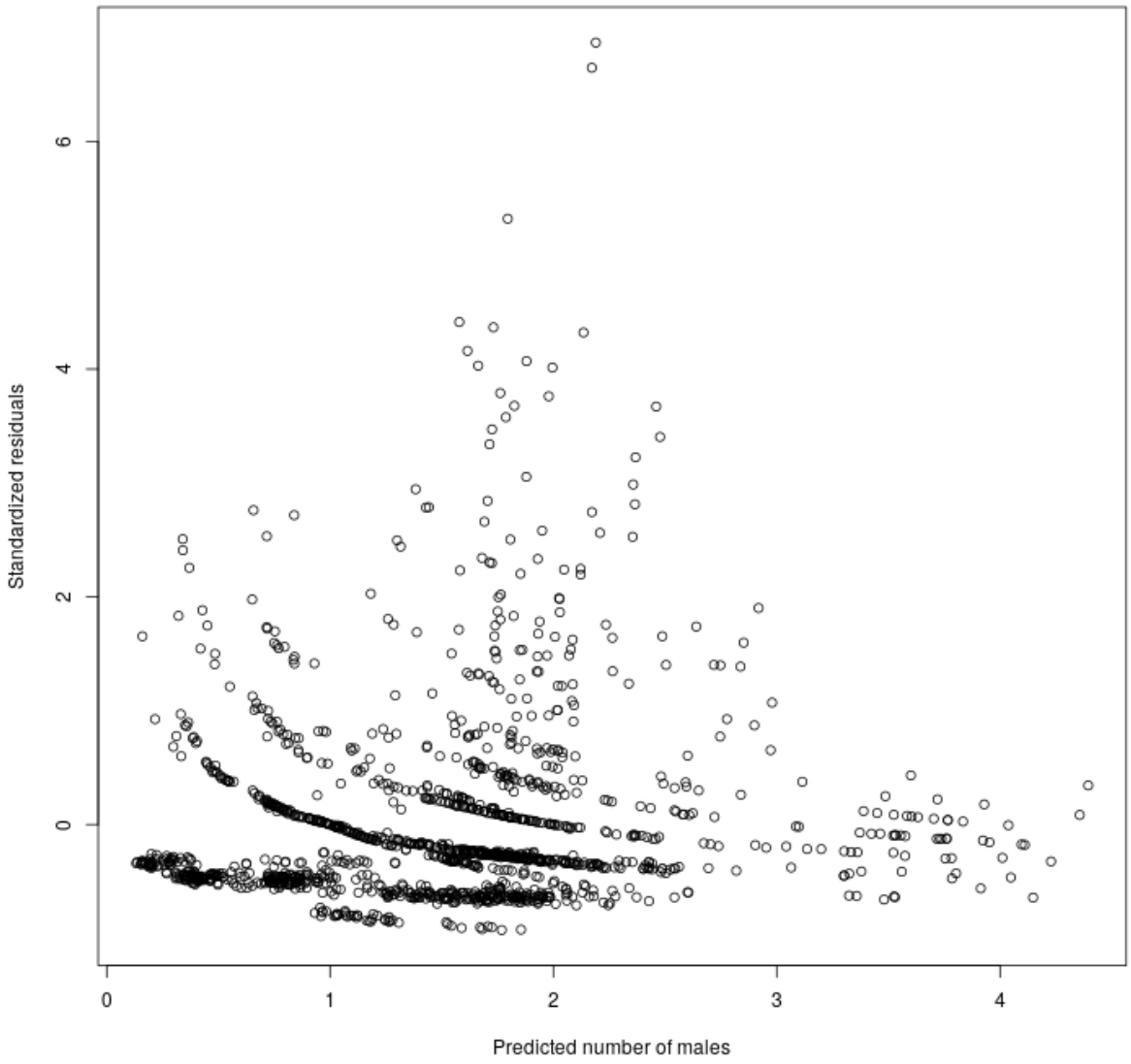
2.1.4 Goodness of fit

We then check the goodness of fit of the model. We simulate M virtual datasets (one per MCMC sample) and consider several summary statistics (see below). For each one, we compare the observed statistics (calculated on the actual dataset) with the statistical distributions of the simulated values. We first use the function `simulateModelCount` to simulate the datasets. WARNING!!! THIS CALCULATION CAN BE VERY LONG (about half an hour). Note that we have included the results of this calculation as a dataset of the package, so that the reader does not need to launch this function to reproduce further calculations:

```
simBinREY <- simulateModelCount(coefModelCountDetectBinREY, dataList)
```

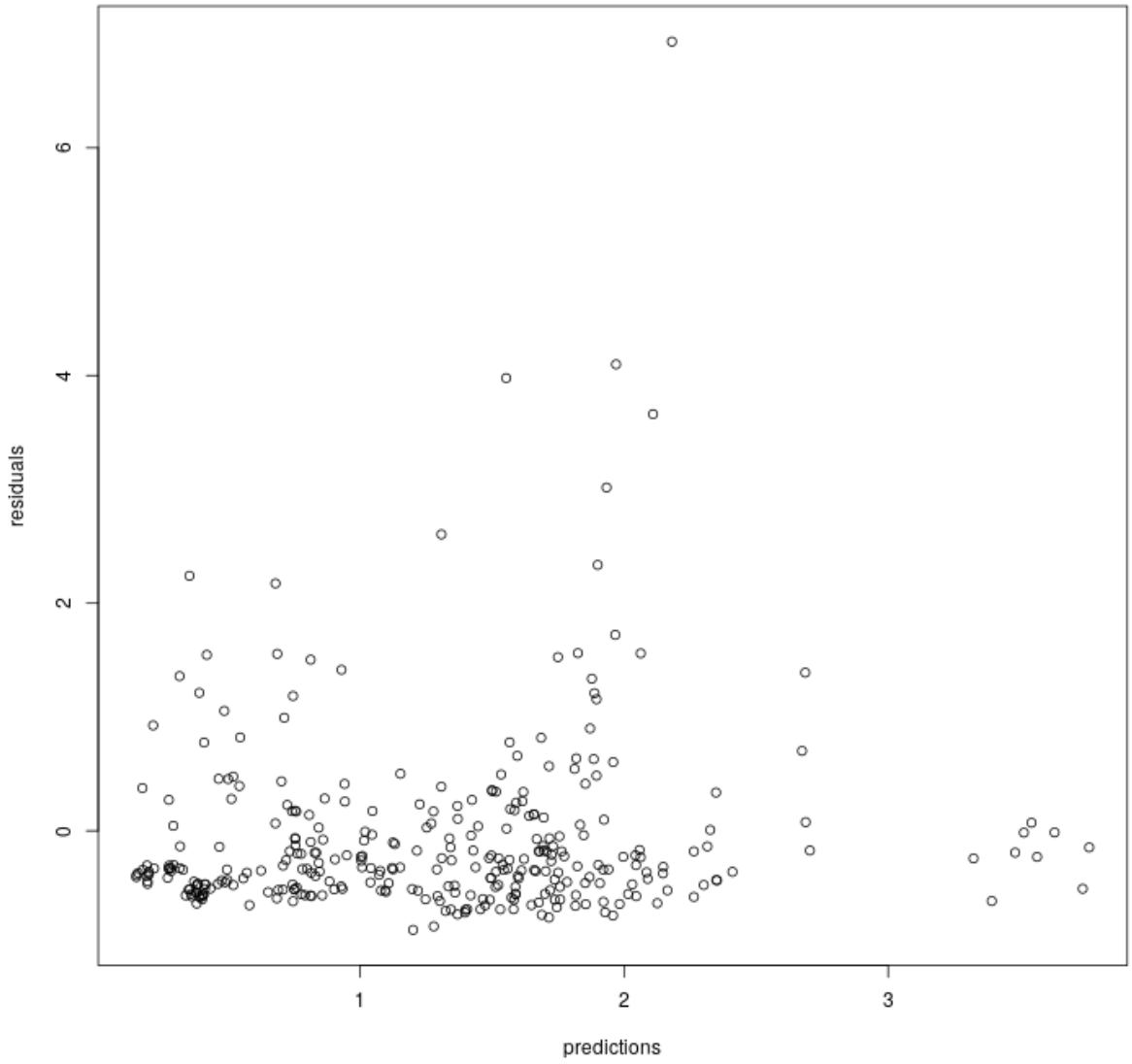
We can then examinate the distribution of the residuals of the model:

```
plot(predict(simBinREY), residuals(simBinREY),
     xlab="Predicted number of males",
     ylab="Standardized residuals")
```



Here, each point corresponds to a census. These residuals do not present any problematic pattern, except a very small number of standardized residuals greater than 6. Note that the group of censuses with a residual greater than 6 correspond to the same lek. Indeed, this appears clearly if we calculate the residuals at the lek level:

```
plot(predict(simBinREY, groupingFactor="lek"),
     residuals(simBinREY, groupingFactor="lek"),
     xlab="predictions",
     ylab="residuals")
```



Here, each point corresponds to one lek. The largest residual correspond to the lek 281, a UL that was discovered in 2013 and censused only this year. We study more in details in section 4 the influence of this lek on the estimation.

2.1.5 Test of goodness of fit

We calculate various statistics to test the goodness of fit of our model; the first statistics is the mean number of males $\hat{y}_{itk}$ for each census, averaged across all our simulated datasets. We can then calculate the chi-square statistics for the observed dataset:

$$\chi^2 = \sum_{i,t,k} \frac{(y_{itk} - \hat{y}_{itk})^2}{\hat{y}_{itk}}$$

We can then compare the observed χ^2 to the distribution of this statistics under the simulated datasets. We calculate below the proportion of simulated χ^2 lower than the observed value:

```r
od <- simBinREY$origData
sim <- simBinREY$sim
ry <- od$nbmales

## Chi2 obs
chi2obs <- (sum(((ry-apply(sim,1,mean))^2)/apply(sim,1,mean)))

## distribution of simulated Chi2
chisq <- ((sim-apply(sim,1,mean))^2)/apply(sim,1,mean)
ch <- (colSums(chisq))

## proportion of simulated chi-square values > observation
mean(ch>=chi2obs)

## [1] 0.67125
```

The observed chi-square is at the center of the simulated statistical distribution.

We also compared the observed total number of detected males over the 10 years of the study with the distribution of simulated values:

```r
mean(colSums(sim)>=sum(ry))

## [1] 0.317
```

...which is also at the center of the simulated statistical distribution. We calculated a 80% credible interval on the expected detected number of males (using the simulated datasets), and computed the proportion of censuses falling in the 80% credible interval:

```r
q1 <- apply(sim,1,quantile,0.1)
q2 <- apply(sim,1,quantile,0.9)
mean((ry>=q1)&(ry<=q2))

## [1] 0.9150103
```

...also indicating a good fit.

We define below a function named `comparef`, which calculates the proportion of simulated numbers greater than or equal to the observed number, for each level of a variable of the observed dataset `od`:

```r
comparef <- function(na)
{
    ta <- tapply(ry,na,sum)
    app <- apply(sim,2,function(x) tapply(x,na,sum))
    sapply(1:nrow(app), function(i) mean(ta[i]>app[i,]))
}
```

We used this function to compare observed values with the statistical distribution of simulated values for different levels of an explanatory variable. For example, we assessed the fit within each geographic region (and use as statistic for each one the total number of animals detected in a region over the 10 years of the study):

```
comparef(od$gr)

## [1] 0.56175 0.55800 0.79800 0.55175 0.77400
```

We can also assessed the fit for each level of different combinations of explanatory variables. For example we assessed the goodness of fit for each combination of geographic region, type of lek, and period:

```
comparef(paste0(od$gr,"-",od$type,"-",od$period))

##  [1] 0.50350 0.44475 0.35600 0.32625 0.00000 0.00000 0.39450
##  [8] 0.22300 0.69625 0.00000 0.92525 0.98875 0.67100 0.74275
## [15] 0.31200 0.55750 0.56125 0.85525 0.17600 0.15650 0.54650
## [22] 0.57025 0.52075 0.66850 0.60300 0.85175 0.50225 0.89850
## [29] 0.65275 0.57175 0.63475 0.55300 0.29725 0.59800 0.74025
## [36] 0.84125 0.81200 0.88525 0.75675 0.35425 0.33600 0.45000
## [43] 0.31900 0.60200 0.23375
```

We have checked the goodness of fit of our model on other variables and combination of variables (leks, gr-years, etc.). We leave it to the reader to play with this function to check other variables if they want.

2.1.6 Identifiability of the parameters

We checked the identifiability of the parameters of interest in our model using the coefficient τ described by [GAR00], which measures the overlap between the prior and the posterior distribution of a parameter. Let x be a parameter of the model, $P(x)$ is the prior distribution for this parameter, and $P(x|\mathbf{D})$ is its posterior distribution. Then, the parameter τ measures the overlap:

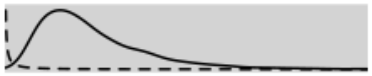
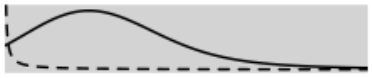
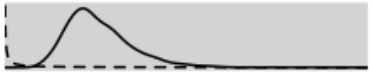
$$\tau = \int_x \min(P(x), P(x|\mathbf{D}))dx$$

This parameter is close to 1 when the dataset $\mathbf{D}$ does not bring a lot of information on the parameter x.

We used the functions `overlapPriorPost` from the package `caperpyogm` (see the help page of this functiond) to calculate the coefficient τ for the variances of random effects in the model, and the function `plotOverlap` from the same package to visualize how the posterior distribution differs from the prior distribution of these parameters:

```
## Use the MCMC samples from the model coefModelCountDetectBinREY
rs <- do.call(rbind, coefModelCountDetectBinREY)

## Detects the name of the parameters for which tau should be calculated
library(stringr)
nam <- c("sigmaREY", "sigmaepsilon","sigmaeta","sigmakappa",
         "sigmanu")
nam <- unlist(lapply(nam, function(z) colnames(rs)[str_detect(colnames(rs),z)]))


## For each parameter, calculates the value of tau:
lb <- sapply(nam, function(na) {
    tau <- overlapPriorPost(coefModelCountDetectBinREY, na,
                            prior="dgamma(den$x[j],0.01, 0.01)", from=0, to=1000, n=10000)
    return(tau)
})

## Builds the data.frame required by plotOverlap
```

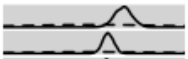
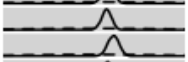
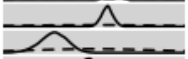
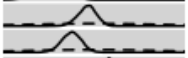
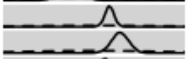
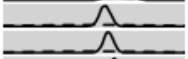
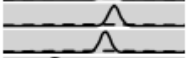
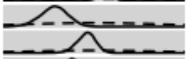
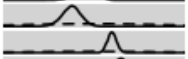
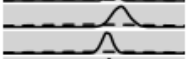
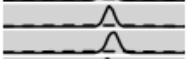
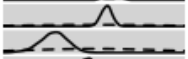
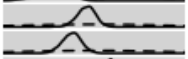
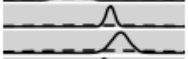
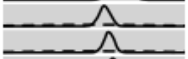
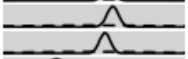
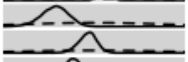
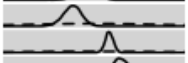
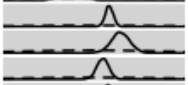
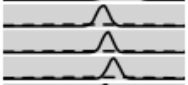
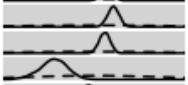
```
df <- data.frame(Parameter=nam, tau=lb)

## Mathematical symbols used for each parameter (see ?plotmath)
namo <- paste0("expression(",  c("sigma[e]", "sigma[epsilon]",
                                 "sigma[eta]^(1)", "sigma[eta]^(2)",
                                 "sigma[eta]^(3)",
                                 "sigma[kappa]","sigma[nu]"),")")
df$namo <- namo

## Prior used for these variances
df$prior <- "dgamma(z,0.01,0.01)"

## customize xy limits for the plots, for a better viz
## of the posterior distribution of each parameter
df$from <- 0.01
df$to <- 10
df$to[1] <- 100
df$to[2] <- 600
df$to[6] <- 600


plotOverlap(coefModelCountDetectBinREY, df, cex=1)
```

Parameter	τ	Distributions
σ_e	0.033	
σ_ε	0.02	
$\sigma_\eta^{(1)}$	0.013	
$\sigma_\eta^{(2)}$	0.067	
$\sigma_\eta^{(3)}$	0.022	
σ_κ	0.033	
σ_ν	0.059	

This table shows the value of the τ parameters and quick plots of the prior (dashed lines) and posterior (solid lines) distribution for each parameter. All the variance parameters are correctly identified, with very low values of τ. We build the same plot for the effects of regions/type of leks/two years period κ_{glb}:

```
## Detects the name of the parameters for which tau should be calculated:
## starting with kappa
nam <- colnames(rs)[str_detect(colnames(rs),"^kappa")]

## note that some of the parameters are not of interest:
## there is only one region for ULs and three regions for KILs.
## (to facilitate calculations, we defined a three way table
##  in JAGS crossing 5 regions, 3 types of leks and 5 periods, but
##  some of these parameters are not used in the model).
## For KILS, only regions 1,2,5 are valid
```

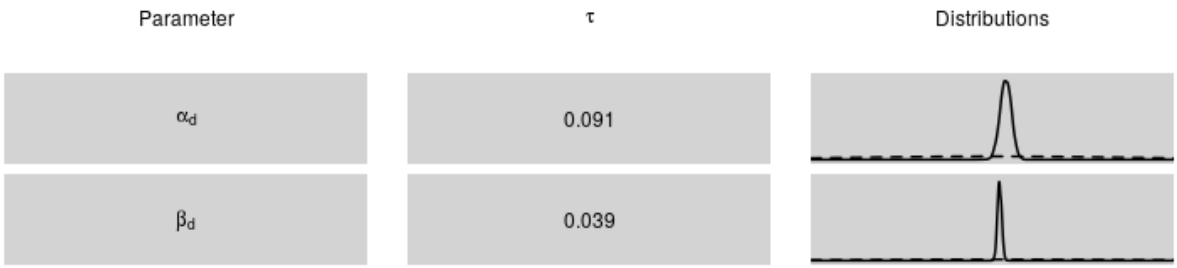
```r
## For ULs, only region 1.
## For KALs, all regions are valid
## We remove unused parameters:
nam <- nam[!(str_detect(nam, "3\\,2\\,.")|
             str_detect(nam, "4\\,2\\,.")|
             str_detect(nam, "[2-5]\\,3\\,."))]

## For each parameter, calculates the value of tau:
lb <- sapply(nam, function(na) {
    tau <- overlapPriorPost(coefModelCountDetectBinREY, na,
                            prior="dnorm(den$x[j],0, sqrt(10))", from=-10, to=10)
    return(tau)
})

## Builds the data.frame required by plotOverlap
df <- data.frame(Parameter=nam, tau=lb)

## Mathematical symbols used for each parameter (see ?plotmath)
namo <- paste0("expression(", gsub(",", "", nam[3:11]),")")
df$namo <- namo

## Prior used for these variances
df$prior <- "dnorm(z,0,sqrt(10))"

## The xy limits for the plots
df$from <- -4
df$to <- 4


plotOverlap(coefModelCountDetectBinREY, df, cex=1)
```

Parameter	τ	Distributions
κ_{311}	0.223	
κ_{411}	0.16	
κ_{511}	0.147	
κ_{121}	0.165	
κ_{221}	0.164	
κ_{521}	0.308	
κ_{131}	0.254	
κ_{112}	0.257	
κ_{212}	0.144	
κ_{311}	0.225	
κ_{411}	0.163	
κ_{511}	0.15	
κ_{121}	0.162	
κ_{221}	0.156	
κ_{521}	0.309	
κ_{131}	0.257	
κ_{112}	0.248	
κ_{212}	0.131	
κ_{311}	0.224	
κ_{411}	0.158	
κ_{511}	0.144	
κ_{121}	0.169	
κ_{221}	0.163	
κ_{521}	0.307	
κ_{131}	0.253	
κ_{112}	0.252	
κ_{212}	0.129	
κ_{311}	0.229	
κ_{411}	0.169	
κ_{511}	0.152	
κ_{121}	0.164	
κ_{221}	0.157	
κ_{521}	0.307	
κ_{131}	0.255	
κ_{112}	0.246	
κ_{212}	0.123	
κ_{311}	0.235	
κ_{411}	0.175	
κ_{511}	0.154	
κ_{121}	0.166	
κ_{221}	0.16	
κ_{521}	0.309	
κ_{131}	0.261	
κ_{112}	0.253	
κ_{212}	0.135	

Similarly, all distributions seem to be correctly identified. Finally, we also present the intercept and slope of the number of observers for the detection model:

```
## Detects the name of the parameters for which tau should be calculated:
## starting with kappa
nam <- c("interceptpd","pente1pd")

## For each parameter, calculates the value of tau:
lb <- sapply(nam, function(na) {
    tau <- overlapPriorPost(coefModelCountDetectBinREY, na,
                            prior="dnorm(den$x[j],0, sqrt(10))", from=-10, to=10)
    return(tau)
})

## Builds the data.frame required by plotOverlap
df <- data.frame(Parameter=nam, tau=lb)

## Mathematical symbols used for each parameter (see ?plotmath)
namo <- paste0("expression(", c("alpha[d]","beta[d]"),")")
```

```
df$namo <- namo

## Prior used for these variances
df$prior <- "dnorm(z,0,sqrt(10))"

## The xy limits for the plots
df$from <- -4
df$to <- 4


plotOverlap(coefModelCountDetectBinREY, df, cex=1)
```



These two parameters are also correctly identified.

2.2 Grid cells search model

2.2.1 Model description

Consider a grid cell q sampled by our protocol. If this cell was sampled before 2018, our protocol implied that observers should search UL in the sampled cell in every case. An update of our grid cells sampling frame in 2018–2019 resulted in the inclusion of additional information in the sampling frame; indeed, several ULs have been discovered between 2010 and 2017 by the network of observers outside the framework of our monitoring (e.g. resulting either from accidental discovery, from compilation of local knowledge, or from local initiatives leading to the search for new leks). See a more detailed explanation of this update of our sampling frame in section 6. Therefore, some of the grid cells sampled in 2018-2019 may already contain a UL discovered previously by the network. In such cases, no search was organized in the sampled grid cell, since the occurrence of a UL was already known.

Let z_q be a binary variable taking a value of 1 if this cell contains a UL detected during a search organized within the framework of our program and a value of 0 otherwise (thus, if a cell q contains a UL discovered prior to its sampling, $z_q = 0$). Let f_q be a binary variable taking a value of 1 if the cell was randomly sampled in 2018 or 2019, and a value of 0 otherwise. Finally, let s_q be a binary variable taking a value of 1 if cell q included a UL discovered prior to its sampling and a value of 0 otherwise.

We defined the following model to describe the probability π_q of UL presence in a grid cell q:

$$\begin{aligned} z_q &\sim \mathcal{B}(b_q \times (1 - s_q) \times \beta) \\ s_q &\sim \mathcal{B}(b_q \times f_q \times \alpha) \\ b_q &\sim \mathcal{B}(\pi_q) \end{aligned}$$

where β is the detection probability during a grid cell search, and α is the probability that a UL present in the cell was discovered prior to its sampling. The parameters α, β, π_q are unknown and must be estimated.

We modeled the probability of presence of a UL in a grid cell with:

$$\text{logit } \pi_q = a_{g(q)} + b_{g(q)} \times r_q + d \times h_q \tag{5}$$

where r_q measures the proportion of the grid cell q covered by area of potential capercaillie presence (defined by experts in 2009, before the onset of the program), and h_q is a binary variable taking a value of 1 if the grid cell q contains a known lek and a value of 0 otherwise. The intercept $a_{g(q)}$ and slope of the area of potential presence $b_{g(q)}$ is supposed to vary between geographical regions.

We estimated the parameter β with the data collected during the lek detection experiment. A sample of grid cells containing a known number of known leks (but unknown to the observers) was drawn and each cell was assigned to one observer. Two observers participated to this experiment: one experienced observer and one inexperienced one. Let N_c be the number of known leks included in the grid cell c. The observer searching this cell defines a search sector. Let S_c be the number of these known leks included in the search sector. Finally, let D_c be the number of known leks detected by the observer in this sector. We define the binary variable $e(c)$ taking a value of 1 if the observer searching the cell c is experienced and a value of 0 if the observer is inexperienced. We used the following model:

$$\begin{aligned} D_c &\sim \mathcal{B}(S_c, \delta) \\ S_c &\sim \mathcal{B}(N_c, \zeta_{e(c)}) \end{aligned}$$

where δ is the probability to detec a lek during the search, given that a lek is present in the search sector, and $\zeta_{e(c)}$ is the probability that the search sector defined by the observer includes a lek. Note that this latter probability depends on the experience of the observer. We supposed that the detection probability β was uniformly distributed between the minimum detection probability $\delta \times \zeta_0$ (characterizing inexperienced observers) and $\delta \times \zeta_1$ (characterizing highly experienced observers):

$$\beta \sim \mathcal{U}(\delta \times \zeta_0, \delta \times \zeta_1)$$

Thus, the unknown parameters of our model are $\{a_{g(q)}\}_{g=1}^3, \{b_{g(q)}\}_{g=1}^3, d, \delta, \zeta_0, \zeta_1$.

2.2.2 Model fit

We need two datasets to fit this model. First, the dataset `gridSearch` contains the data collected during grid cells search:

```
head(gridSearch)
```

```
##   presenceArea newUL previousSearch hasPreviousUL gr
## 1    0.9172760     0              0             0  1
## 2    0.4677633     1              0             0  1
## 3    0.4480399     0              0             0  1
## 4    0.9542109     0              0             0  1
## 5    0.5240671     0              0             0  2
## 6    0.6726517     0              0             0  1
##   presenceKL
## 1          1
## 2          1
## 3          0
## 4          0
## 5          1
## 6          1
```

For each grid cell search, this data.frame contains:

- The proportion of the grid cell covered by the area of potential capercaillie presence (as defined by a group of experts in 2009, before the onset of the program).
- Whether (1) or not (0) a UL was discovered during the cell search.
- Whether (1) or not (0) the cell was sampled in 2018-2019 (i.e. could possibly contain a UL discovered outside the framework of our program, prior to its sampling)
- Whether (1) or not (0) the cell contained a UL discovered prior to its sampling.
- The label of the geographic region containing the grid cell (1= Piemont, 2=High area, 3 = Pyrenees national parc).
- Whether (1) or not (0) the grid cell contained a known lek.

We also need the data.frame `DetectionExpe`, containing the data collected during the experiment designed to assess the detectability of new leks during grid cells searches:

```
DetectionExpe
```

```
##         observer Nquad Nsect Ndete
## 1  Inexperienced     5     2     2
## 2  Inexperienced     2     1     0
## 3  Inexperienced     2     1     0
## 4  Inexperienced     1     1     1
## 5  Inexperienced     2     1     1
## 6  Inexperienced     4     1     1
## 7  Inexperienced     2     0     0
## 8  Inexperienced     1     1     1
## 9  Inexperienced     3     2     1
## 10   Experienced     1     1     1
## 11   Experienced     0     0     0
## 12   Experienced     1     0     0
## 13   Experienced     3     2     1
## 14   Experienced     2     2     2
## 15   Experienced     1     1     1
## 16   Experienced     1     1     0
```

For each grid cell sampled for this experiment, this data.frame contains:

- Whether the observer was experienced or inexperienced.
- The number of known leks in the cell.
- The number of known leks present in the search sector defined by the observer.
- The number of known leks detected by the observer.

We have programmed the model in JAGS:

```
cat(modelULPresence)
```

```
## model {
##
## P_inclusion_inex~dunif(0.0000001, 0.9999999)
```

```
## P_inclusion_expe~dunif(0.0000001, 0.9999999)
## P_detection~dunif(0.0000001, 0.9999999)
##
## for (i in 1:Ninex) {
##     inex_Nsect[i]~dbin(P_inclusion_inex, inex_Nquad[i])
##     inex_Ndete[i]~dbin(P_detection, inex_Nsect[i])
## }
##
##
## for (i in 1:Nexpe) {
##     expe_Nsect[i]~dbin(P_inclusion_expe, expe_Nquad[i])
##     expe_Ndete[i]~dbin(P_detection, expe_Nsect[i])
## }
##
## pmin <- P_detection*P_inclusion_inex
## pmax <- P_detection*P_inclusion_expe
## pd ~ dunif(pmin, pmax)
##
##
## pdprev~dunif(0,1)
## for (i in 1:Ngr) {
##   a[i]~dnorm(0,0.1)
##   b[i]~dnorm(0,0.1)
## }
## d~dnorm(0,0.1)
##
## for (i in 1:Nq) {
## logit(pp[i]) <- a[gr[i]]+b[gr[i]]*presenceArea[i]+d*presenceKL[i]
## pres[i]~dbern(pp[i])
## hasPreviousUL[i]~dbern(pres[i]*previousSearch[i]*pdprev)
## newUL[i]~dbern(pres[i]*(1-hasPreviousUL[i])*pd)
## }
##
## }
```

We prepare the datasets for the fit:

```
dataListQ <- datagrid2jags(gridSearch, DetectionExpe)
```

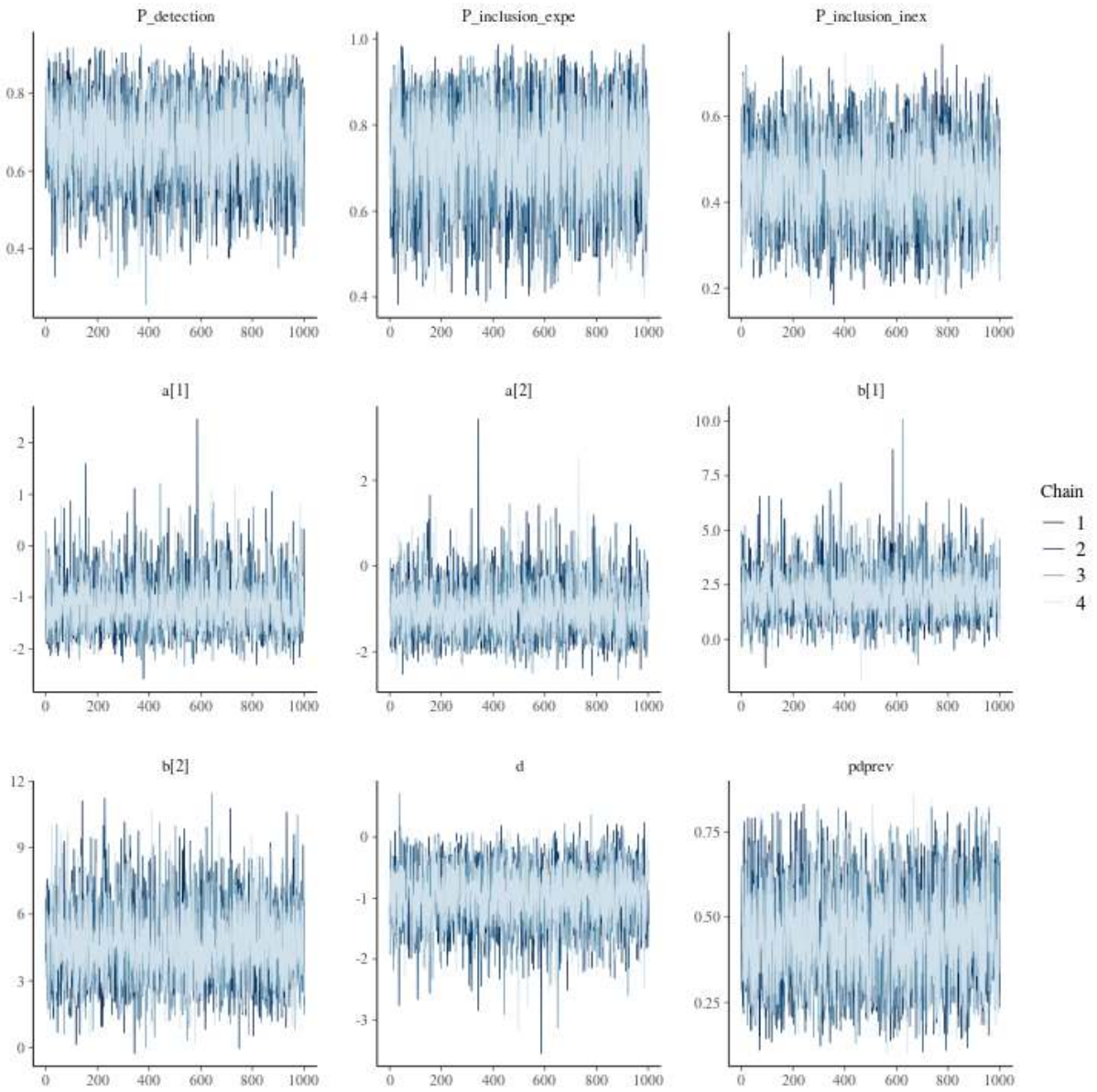
We can now use the function `fitModelGrid` to fit this model. WARNING: THIS CALCULATION TAKES A VERY LONG TIME (several hours) !!!! Note that we have included the results of this calculation as a dataset of the package, so that the reader does not need to launch this function to reproduce further calculations:

```
coefModelULPresence <- fitModelGrid(dataListQ, "modelULPresence")
```

2.2.3 MCMC chains mixing

Once the MCMC samples have been obtained, we can plot the chains for a visual examination of the chain mixing:

```
mcmc_trace(coefModelULPresence)
```



We also calculate the diagnostic of Gelman and Rubin (1992) for the parameters:

```
gelman.diag(coefModelULPresence)
```

Mixing properties are excellent here.

2.2.4 Goodness of fit

We then check the goodness of fit of the model. As for the count model, we simulate M virtual datasets (one per MCMC sample) and consider several summary statistics (see below). For each one, we compare the observed statistics (calculated on the actual dataset) with the statistical distribution of simulated values. We first use the function `simulateModelGrid` to simulate the datasets. Fortunately, this function is faster than the one used for the count model, so that it can readily be executed by the user:

```
sg <- simulateModelGrid(coefModelULPresence, dataListQ)
```

We can then calculate the proportion of simulated values lower than the observed value of the total number of ULs discovered in our study:

```
obs <- sum(dataListQ$newUL)
sim <- colSums(sg$sim)
mean(sim<obs)

## [1] 0.4855
```

The observed value is at the center of the distribution expected under our model. We can calculate the same proportion for each geographical region (piemont or high mountains):

```
obs <- tapply(dataListQ$newUL, dataListQ$gr, sum)
sim <- apply(sg$sim,2,function(x) tapply(x, dataListQ$gr, sum))

## Region 1
mean(sim[1,]<obs[1])

## [1] 0.56475

## Region 2
mean(sim[2,]<obs[2])

## [1] 0.34575
```

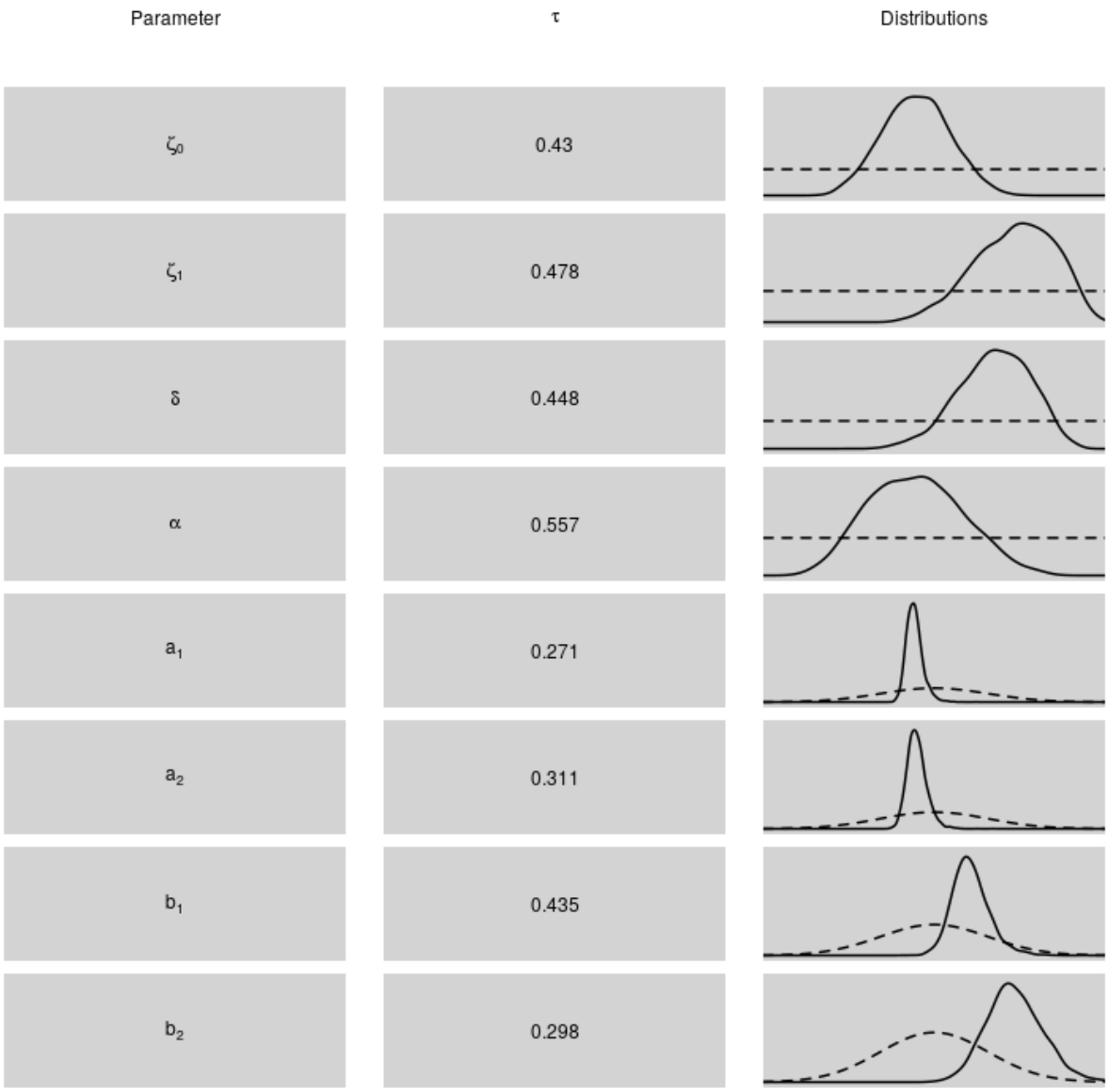
The observed values are located within the distribution of simulated values.

2.2.5 Identifiability of the parameters

We also assessed the identifiability of the parameters of interest in this model, using the parameter τ described by [GAR00] (see section 2.1.6):

```
rs <- do.call(rbind, coefModelULPresence)
nam <- c("P_inclusion_inex","P_inclusion_expe","P_detection",
         "pd$","pdprev$","a","b")
nam <- unlist(lapply(nam, function(z) colnames(rs)[str_detect(colnames(rs),z)]))


## For each parameter, calculates the value of tau:
lb <- sapply(nam, function(na) {
    if (str_detect(na, "^a")|str_detect(na,"^b")) {
        tau <- overlapPriorPost(coefModelULPresence, na)
    } else {
        tau <- overlapPriorPost(coefModelULPresence, na,
                                prior="dunif(den$x[j],0, 1)", from=0, to=1)
    }
    return(tau)
})

## Builds the data.frame required by plotOverlap
df <- data.frame(Parameter=nam, tau=lb)
namo <- paste0("expression(", c("zeta[0]","zeta[1]","delta","alpha", nam[5:8]),")")
df$namo <- namo
df$prior <- "dunif(z,0,1)"
df$prior[5:8] <- "dnorm(z,0,sqrt(10))"
df$from <- 0
df$to <- 1
df$from[5:8] <- -10
```

```
df$to[5:8] <- 10


plotOverlap(coefModelULPresence, df, cex=1)
```



The parameters of this model were not as precisely identified as for the model of counts. The probability α is the less precisely identified parameter; however this parameter is not a parameter of interest: we included this parameter in our model to account for the fact that we should not model the detection process during search when it was known that a lek was present in a grid cell. Thus, the weak identifiability for this parameter is not a problem. Other parameters are more precisely identified, though there is still a large imprecision for many of them – in particular the parameters estimated using our field experiment. Future data collection will allow to precise these posterior distributions, but meanwhile we will accept these estimates.

2.3 Estimation of the number of males in each region

Let $S_{g\ell}$ be the number of leks of type ℓ in region g. Note that this number needs to be estimated for ULs (i.e. for $\ell = 3$). The grid cell search model can be used to estimate S_{g3}. We estimated this number with:

$$\widehat{S_{g3}} = \sum_{q=1}^{Q} \pi_q \tag{6}$$

where the probability π_q is calculated for each grid cell q with the equation 5, and the sum is calculated over all the grid cells of the sampling frame. Then, the number of males $n_{g\ell b}$ in geographic region g for lek type ℓ during period b can be estimated with:

$$\widehat{n_{g\ell b}} = \sum_{\ell=1}^{3} S_{g\ell} \exp\left\{\kappa_{g,\ell,b} + I(\ell = 2) \times \sigma_\nu^2/2 + \sigma_\eta^2/2 + \sigma_\epsilon^2/2\right\}$$

Of course, we can calculate one estimate $\widehat{n_{g\ell b}}$ per MCMC iteration, so that the posterior distribution of the number of males in a given region during a given period can be readily obtained. Finally, the number of males over the whole mountain range during a given period b is calculated by:

$$\widehat{n_b} = \sum_{g=1}^{5} \sum_{\ell=1}^{2} \widehat{n_{g\ell b}}$$

We store the number of KAL and KIL in each one of the five regions in R vectors:

```
NKAL <- c(14L, 64L, 76L, 48L, 46L)
NKIL <- c(6L, 80L, 102L, 32L, 96L)
```

We use the function `estimateNmales` to combine the two models and estimate the number of males in the different geographical regions. The frame of grid cells as an argument is required for the estimation of the number of ULs in each region. This frame is stored in the dataset `gridFrame`:

```
head(gridFrame)
```

```
##   presenceArea presenceKL gr grqf
## 1  -0.40143980          0  2    2
## 2  -0.33081494          0  2    2
## 3  -0.40143980          0  2    2
## 4  -0.33839617          0  2    2
## 5  -0.35121600          0  2    2
## 6   0.03079977          0  2    2
```

This data.frame contains the following variables for each grid cell:

- proportion of the cell covered by the potential capercaillie presence area
- whether the cell contains (1) or not (0) an already known lek
- the geographic region (original partitioning in 5 regions)
- the geographic region code used in the grid cell search model (1 = piemont, 2 = high range)

```
nm <- estimateNmales(coefModelCountDetectBinREY, coefModelULPresence,
                     gridFrame, NKAL, NKIL)
```

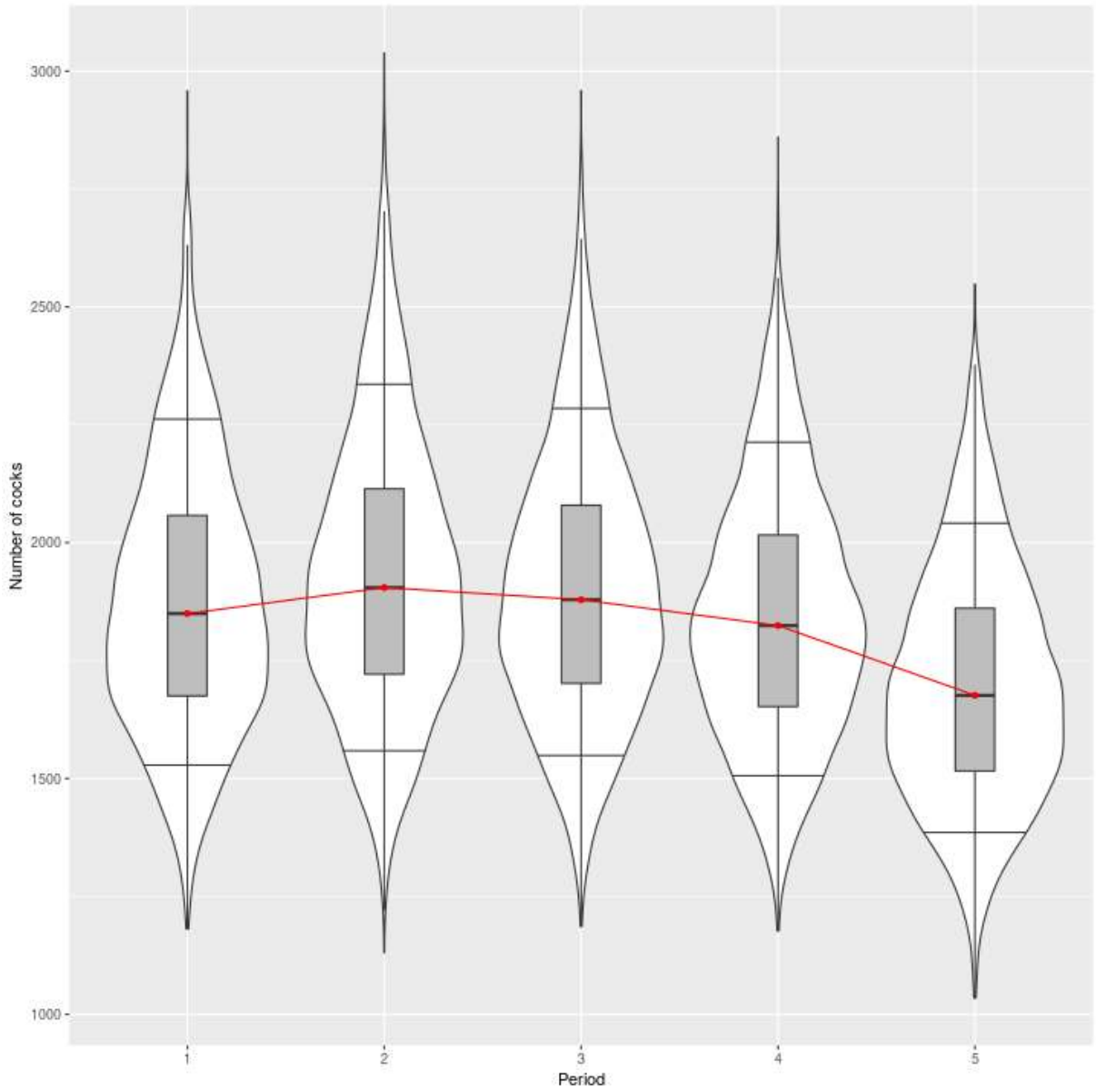
The function `summary` shows the estimated number of males for all two-year periods:

```
summary(nm)

## Estimates of the number of males during the 5 periods
## in the whole Pyrenees Mountains  (median and 80% CI):
##
##   Point.est    CIlow     CIup
## 1  1871.789 1542.350 2350.656
## 2  1922.285 1570.448 2440.920
## 3  1901.102 1562.484 2387.119
## 4  1841.119 1516.075 2308.645
## 5  1694.132 1398.089 2129.504
##
## Change rate between period 1 and period 5 is: -8.49% (80% CI: -20.24% -- 0.13%)
##
## Probability of scenarii (decrease:< -10%; stability:-10--10%; increase:>10%):
##
##    scenario probability
## 1  decrease     0.43100
## 2 stability     0.56575
## 3  increase     0.00325
```

This function returns the point estimate of the population size each two-year period (with a 80% credible interval), the point estimate of the rate of change of this size between the first and last period, and the probability of the three scenarios: increasing population (actual change rate $> 10\%$), decreasing population (actual change rate $< -10\%$), stability (otherwise).

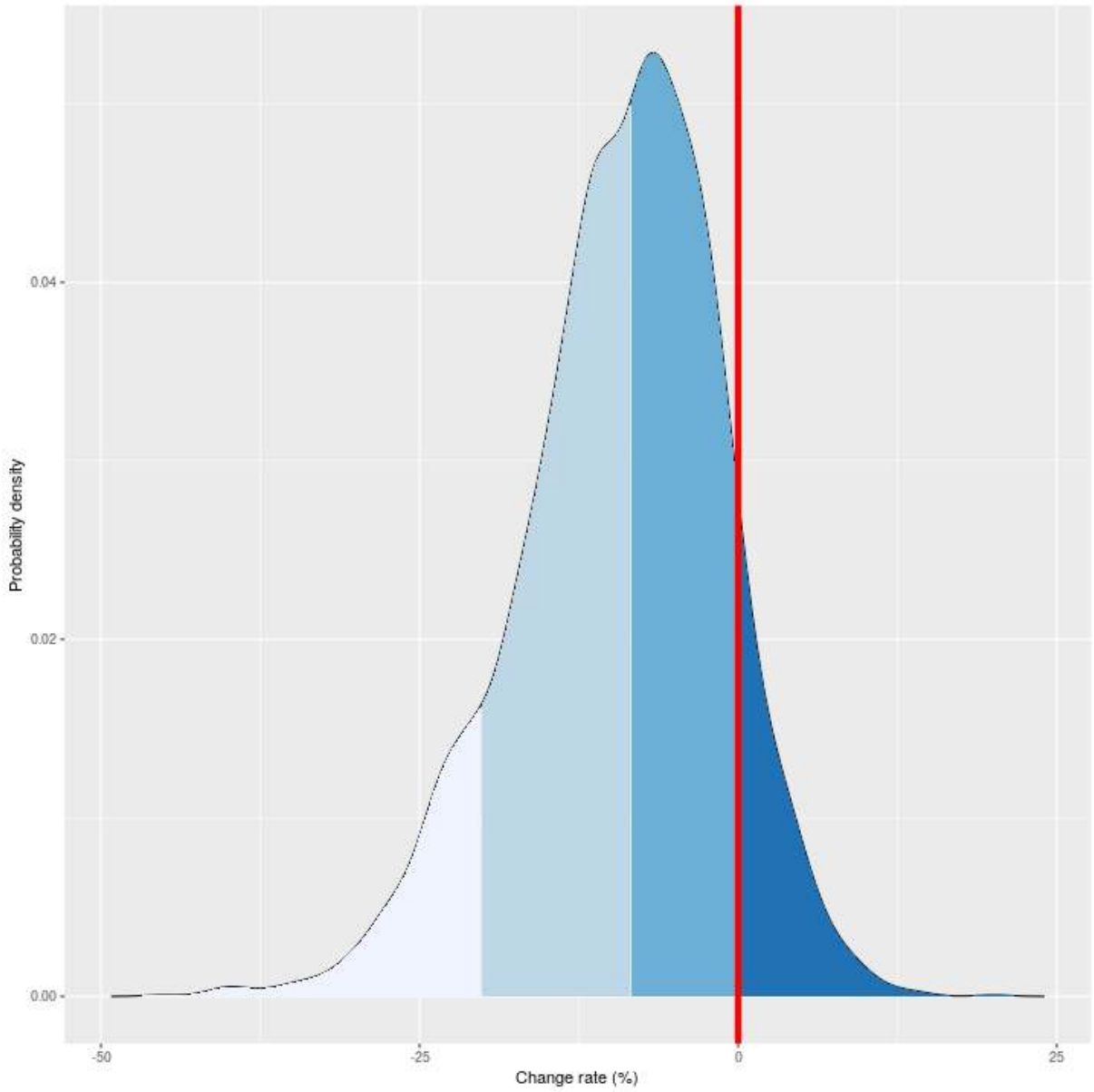
The function `distritime` shows the posterior distribution of the estimated population size during the 5 periods:

```
distritime(getTotal(nm))
```



The function `showChangeRate` can be used to estimate the rate of change of the population size between the first and last two-year periods:

```
showChangeRate(nm)
```



Note that we can estimate the number of males in a given region with the function `summary`. For example, for the region 1:

```
summary(nm,1)

## Estimates of the number of males during the 5 periods
## in the geographic region 1  (median and 80% CI):
##
##   Point.est     CIlow     CIup
## 1  155.9515 111.19115 227.6323
## 2  154.9948 109.65762 219.2993
## 3  149.0112 106.98677 213.7739
## 4  145.4841 104.07529 209.1013
## 5  135.4594  95.31548 198.5411
##
## Change rate between period 1 and period 5 is: -12.1% (80% CI: -28.59% -- 2.13%)
##
## Probability of scenarii (decrease:< -10%; stability:-10--10%; increase:>10%):
##
```

```
##    scenario probability
## 1  decrease     0.57125
## 2 stability     0.40050
## 3  increase     0.02825
```

3 Assumptions on the detection process and on constant number of males within two-years periods

3.1 List of alternative models

The population size estimation by a N-mixture model can be very sensitive to the misspecification of the detection process [LIN18]. We considered several alternative models for this process before choosing the approach detailed in our paper. We also tried to check our assumption of a constant number of males within the two-year periods. We describe these alternative models in this section, and show in the next sections how we compared the different models with cross-validation.

We recall the detection model used in the paper:

$$\text{logit } p_{i,t,k} = \alpha_d + \beta_d \times o_{i,t,k} + e_{g(i),t}$$

where $o_{i,t,k}$ is the number of observers partipating to k-th census on lek i during year t, and $e_{g(i),t}$ is a Gaussian random effect describing the variation in detectability during a given year in a given region. As already described, this model is programmed for the JAGS software in the dataset `modelCountDetectBinREY` of the package.

We first tested whether the year random effects were required in the model, by fitting the simpler model:

$$\text{logit } p_{i,t,k} = \alpha_d + \beta_d \times o_{i,t,k}$$

i.e., the same model as before, but without the year random effects. This model is programmed for the JAGS software in the dataset `modelCountDetectBin`.

We also tested the model `modelCountDetectBin` by considering one-year periods. Indeed, in our paper, we have supposed that the number of males present in a given region did not vary within two-year periods. Within a given period, the between-year variation in the number of detected males was supposed to be caused by between-year variability in the detection probability. We therefore tried to check this assumption by setting $b(t) = t$ in equations 1, 3, and 4, and by fitting the binomial detection model without the random effects of the years.

We also tested whether an extra-binomial variation was required for the detection process of our original model (i.e. with two-year periods and year random effects), by modeling the number of detected animals as a beta-binomial distribution (see [MAR11]). That is, we considered the following model:

$$\begin{aligned} \text{logit } d_{i,t,k} &= \alpha_d + \beta_d \times o_{i,t,k} + e_{g(i),t} \\ p_{i,t,k} &\sim \text{Beta}\left(d_{i,t,k} \times \frac{1-\delta^2}{\delta^2}, (1-d_{i,t,k}) \times \frac{1-\delta^2}{\delta^2}\right) \end{aligned}$$

This model is programmed for the JAGS software in the dataset `modelCountDetectBetaBinREY`.

Finally, we tested whether the relationship between the detection probability and the number of observers was linear by including a quadratic effect of this variable in our model:

$$\text{logit } p_{i,t,k} = \alpha_d + \beta_d \times o_{i,t,k} + \omega_d \times o_{i,t,k}^2$$

This model is programmed for the JAGS software in the dataset `modelCountDetectBinREYObs2`.

We compared these models using cross-validation. We describe in the next subsections how we carried out this comparison.

3.2 On the use of cross-validation to compare several models. Theory

We have used the approach described by [VEH17] to compare the different models for the detection process. This approach relies on cross-validation and identifies, in a set of alternative models, the model that return the best predictions of new data. We describe the rationale behind this approach in this section, and we show how we used it in the next section.

Consider two models $\mathcal{M}_0$ and $\mathcal{M}_1$ fitted to the same dataset. These two models can be compared with the *expected log pointwise predictive density for a new dataset*, or **elpd**. First consider a simple observed dataset with n statistical units, where each unit i is characterized by a value of a response variable y_i. For example, this dataset may contain the numbers y_i of detected cocks during each census i, among n censuses carried out on a unique lek. We measure the quality of prediction by a model of the data by calculating the elpd of the model, i.e. by trying to predict the number of detected males in n new censuses of the same lek at the same year, in the same conditions.

In theory, for a given model – say $\mathcal{M}_0$ – this criterion should be calculated by:

$$\text{elpd} = \sum_{i=1}^{n} \int p_t(\tilde{y}_i) \log p(\tilde{y}_i|y, \mathcal{M}_0) d\tilde{y}_i$$

where $\tilde{y}_i$ is the value of the variable y for unit i in the new dataset (in our example, the number of detected males during census i on the lek in our new dataset). The value of $p_t(\tilde{y}_i)$ is the *true* probability to obtain $\tilde{y}_i$ (under the true process), and $p(\tilde{y}_i|y, \mathcal{M}_0)$ the probability to obtain $\tilde{y}_i$ under the model $\mathcal{M}_0$ fitted to the observed dataset y. When the elpd criterion is high, the model closely describes the reality.

In practice, we cannot calculate the value of the elpd since we do not know the true model $p_t(\tilde{y}_i)$. [VEH17] propose several approaches relying on cross-validation to estimate the elpd with the available dataset; we describe here the *K-fold cross validation* approach that we implemented in our study.

To implement this approach, we must randomly partition the dataset in K subsamples. At each step of the cross-validation process, one subsample is retained as a validation set and the remaining $K-1$ subsamples are pooled together and used as a calibration set. The model $\mathcal{M}_0$ is then fitted to the calibration set using MCMC, leading to S vectors $\theta^{s,k}$ randomly drawn from the posterior distribution $p(\theta|y_{-k}, \mathcal{M}_0)$, with y_{-k} the calibration dataset obtained after the removal of the subsample k. In other words, $\theta^{s,k}$ is the θ simulated at the iteration s of the MCMC used to fit the model after removal of the subsample k. Then, for each observation i belonging to the validation set k, we calculate the contribution of the observation i to the estimated elpd by:

$$\widehat{\text{elpd}}_i(\mathcal{M}_0) = \log \left(\frac{1}{S} \sum_{s=1}^{S} p(y_i|\theta^{k,s}, \mathcal{M}_0) \right)$$

with $p(y_i|\theta^{k,s}, \mathcal{M}_0)$ the probability to obtain the observed value y_i, for the parameter vector $\theta^{s,k}$ generated by the s-th MCMC iteration. This probability can be calculated using the model $\mathcal{M}_0$.

This operation is repeated on all subsamples k, which allows to estimate the contribution $\widehat{\text{elpd}}_i$ to the elpd for all observations i of the dataset. We can estimate the total elpd by:

$$\widehat{\text{elpd}}_{\text{xval}}(\mathcal{M}_0) = \sum_i \widehat{\text{elpd}}_i(\mathcal{M}_0)$$

The same operation can be repeated for model $\mathcal{M}_1$, and the model with the largest value of $\widehat{elpd}_{xval}$ is the model with the best predictive efficiency.

3.3 K-fold cross-validation in our study

Now consider the N-mixture model used to predict the mean number of capercaillie cocks on leks in our study. This is a hierarchical model, which raises the question of the "level" at which the statistical unit should be defined to assess the predictive ability of our model (the census? the set of censuses on a lek during a given year? the whole set of censuses carried out during all years of the study period on a given lek? etc.). We must consider the aim of the model to choose this level appropriately [MER19].

Actually, the estimation of the capercaillie population sizes and trends strongly relies on our ability to predict the time series of the number of males on leks not included in the sample. Thus, we must include *all* censuses carried out on a lek (all censuses and all years) to the same subsample, and assess the predictive efficiency at the level of the lek. This approach is the *grouped K-fold for leave-one-group out* defined in `https://avehtari.github.io/modelselection/rats_kcv.html`. In other words, we have N leks (each one being censused one or several years, with one or several censuses every year), and these N leks are partitionned in K subsamples of N/k leks. We partitionned our dataset in $K = 10$ subsamples ([VEH17] indeed recommend to define at least 10 subsamples in *K-fold cross validation* approaches).

As described in the previous section, we fitted a model on a calibration dataset excluding each subsample k in turn. For each calibration dataset excluding a subsample k, we then drew a MCMC sample of S vectors of parameters $\theta^{k,s}$ containing the top parameters of the model $(\kappa_{g(i),\ell(i),b(t)}, \sigma_\kappa, \sigma_\epsilon, \sigma_\eta, \sigma_\nu, \alpha_d, \beta_d, \sigma_e)$.

We assess for each lek of the group k, the contribution to the elpd of the set of censuses carried out on lek i. Thus, if $\mathbf{y}_i$ is the vector containing the set of censuses carried out on lek i between 2010 and 2019, we need to calculate the probability $p(\mathbf{y}_i|\theta^{k,s}, \mathcal{M}_0)$ to get this set of censuses for each vector $\theta^{k,s}$. The contribution of the lek i to the elpd is calculated by:

$$\widehat{\text{elpd}}_i(\mathcal{M}_0) = \log\left(\frac{1}{S}\sum_{s=1}^{S} p(\mathbf{y}_i|\theta^{k,s}, \mathcal{M}_0)\right)$$

And the total elpd is calculated by summing these contributions over all leks.

The calculation of $p(\mathbf{y}_i|\theta^{k,s}, \mathcal{M}_0)$, required for the calculation of elpd, is a bit tricky. Indeed, it can be calculated by:

$$p(\mathbf{y}_i|\theta^{k,s}, \mathcal{M}_0) = \int p(\mathbf{r}_i|\mathbf{N}_i) \times p(\mathbf{N}_i|\theta^{k,s}, \mathcal{M}_0) d\mathbf{N}_i$$

with $\mathbf{N}_i$ the vector containing the actual numbers of males on the lek i for each two-year period. We approximate this integral by a Monte Carlo approach [ROB10]. For each vector $\theta^{k,s}$ generated by MCMC, we simulated $R = 1000$ vectors $\mathbf{N}_i^{(r)}$ using our model. For each simulated vector: (i) we simulated for each a lek effect η_i, an overdispersion residual ϵ_{ij}, and possibly a natural unit effect $\nu_{u(i)}$; (ii) we summed these effects and added the simulated value of $\kappa_{g(i),\ell(i),b}$ to calculate the value of $\log \lambda_{it}^{(r)}$; (iii) we randomly sampled R actual numbers of males $N_{it}^{(r)}$ in a Poisson distribution with parameter $\lambda_{it}^{(r)}$), and we derived from each vector $\mathbf{N}_i^{(r)} = \{N_{it}^{(r)}\}_{p=1}^{5}$ the probability $p(\mathbf{y}_i|\mathbf{N}_i^{(r)}, \theta^{k,s})$ under our detection model. The probability $p(\mathbf{y}_i|\theta^{k,s}, \mathcal{M}_0)$ is then equal to the mean over all R simulations of these probabilities $p(\mathbf{y}_i|\mathbf{N}_i^{(r)}, \theta^{k,s})$:

$$\hat{p}(\mathbf{y}_i|\theta^{k,s}, \mathcal{M}_0) = \frac{1}{R}\sum_{r=1}^{R} p(\mathbf{y}_i|\mathbf{N}_i^{(r)}, \theta^{k,s})$$

The probability $p(\mathbf{y}_i|\mathbf{N}_i^{(r)}, \theta^{k,s})$ is calculated by the product of the probabilities $p(y_{itk}|N_{it}, \theta^{k,s})$ over all censuses carried out during our study period.

To sum up, *in theory*, for a given model $\mathcal{M}_0$, the elpd is calculated by summing the individual contributions $\widehat{\text{elpd}}_i(\mathcal{M}_0)$ over all leks i. However, the number of censuses differs a lot between leks (between 1 to 19 censuses in 10 years). However, the probability to observe a given set of censuses decreases when the number of censuses increases: for example, if p is the probability to detect n males during a census, then the probability to detect n males during each one of two censuses is p^2 if these two censuses are independent. In other words, the logarithm of the mean of $p(\mathbf{y}_i|\theta^{k,s}, \mathcal{M}_0)$ over all simulations s is much smaller for leks with many censuses: these leks have a much larger weight in the calculation the elpd.

This problem was not considered in [VEH17] (who focus their paper on leave-one-out cross-validation, so that there is no difference in sample size among statistical units in their study). We propose to complete this first calculation of the elpd with another approach attempting to correct these unequal weights. As noted before, the order of magnitude of the probability to observe n events is p^n if the probability to observe one event is p and if these events are independent. Therefore, the log-probability will be equal to $n \log p$.

Therefore, *if the c_i censuses carried out on a given lek i were independent*, we would expect $\widehat{\text{elpd}}_i(\mathcal{M}_0)$ to decrease linearly with c_i. In such a case, replacing $\widehat{\text{elpd}}_i(\mathcal{M}_0)$ in the calculation of the elpd with $\widehat{\text{elpd}}_i(\mathcal{M}_0)/c_i$ would allow to give the same weight to all leks in the calculation of the elpd. Let eldp^c be the elpd "corrected" by this approach.

However, in our study, the c_i censuses carried out on a given lek *are not* independent: some censuses are carried out during the same year and some during different years; some are carried out during the same period and some during different periods, etc. Therefore, c_i is an overestimation of the "effective" sample size for lek i, and dividing the $\widehat{\text{elpd}}_i(\mathcal{M}_0)$ by c_i will "over-correct" the places with the largest number of censuses.

Thus, elpd gives too much weight to the big leks in the model comparison, and elpd^c does not give enough weight to these leks. However, when these two criteria return the same conclusions, we can be confident that the selected model is the best one in the set of compared models.

We calculated the contributions to elpd and elpd^c for each model. We also calculated the standard deviation of the difference between the elpd/elpd^c of a model $\mathcal{M}_r$ and the elpd/elpd^c of the best model $\mathcal{M}_b$ in the set. This standard deviation is calculated by:

$$\sqrt{\frac{n}{n-1}\sum_{i=1}^{n}(\Delta_i - \bar{\Delta}_i)^2}$$

Where $\Delta_i = \widehat{\text{elpd}}_i(\mathcal{M}_r) - \widehat{\text{elpd}}_i(\mathcal{M}_b)$, and $\bar{\Delta}$ is the mean of the Δ_i. The standard deviation for the elpd^c is obtained similarly by replacing elpd by elpd^c in the above formula.

Finally, we carried out a randomization test to determine if the elpd of a model was significantly different from the elpd of the best model. We used the following test criterion:

$$T = \sum_i \Delta_i$$

which is the difference between the elpd of the model and the elpd of the best model. We then carried out a randomization test, changing randomly the sign of the Δ_i's and calculating again the criterion. We repeated this randomization 1000 times, and calculated the proportion of simulated values for T greater than the observed value.

3.4 Implementation in R

The K-fold cross validation approach is implemented in the function `kfoldCVModelCount`. Our dataset contains 330 leks, so that we define a partition of 10 groups of 33 leks:

```
set.seed(980)
ooo <- sample(c(rep(1:10,each=33)))
```

Then, we can use the function `kfoldCVModelCount` to implement cross-validation. We illustrate the process below for the model `modelCountDetectBinREY` used in our paper. WARNING!!! THIS CALCULATION CAN TAKE SEVERAL HOURS. Note that we have included the results of this calculation as a dataset of the package, so that the reader does not need to launch this function to reproduce further calculations:

```
listCoefsCVBinREY <- kfoldCVModelCount(ooo, dataList, "modelCountDetectBinREY")
```

The resulting object `listCoefsCVBinREY` is a list containing the values of the coefficients sampled by MCMC by excluding each subsample of leks in turn (the first element of the list contains the coefficients sampled by MCMC when fitting the model on a dataset excluding the group 1 of lek, etc.). We can then use the function `LLCount` to calculate the matrix containing the probabilities $p(\mathbf{y}_i|\mathbf{N}_i^{(r)}, \theta^{k,s})$ to observe the set of detected males during all censuses for a given lek (rows) under the model with a given sampled vector of parameters (columns). WARNING, this calculation can also be very long. The result of this calculation is also available as a dataset in the package:

```
llcBinREY <- LLCount(dataList, listCoefsCVBinREY, ooo)
```

Finally, we can calculate the contribution of each lek to the elpd:

```
elpdBinREY <- elpdLeks(llcBinREY)
```

The same approach is used for other models. Note that because the lists of coefficients sampled by MCMC returned by the function `kfoldCVModelCount` are very large objects ($>$ 40 MB), we do not include such objects for other models in our package. However, the result of the function `LLCount` is presented for all of them. For the record, the code used to obtain such object, WHICH TAKES SEVERAL DAYS OF COMPUTING, is presented below. The resulting objects (with names starting by `llc`) are stored as datasets of the package:

```
## K-fold CV for different models
listCoefsCVBin <- kfoldCVModelCount(ooo, dataList, "modelCountDetectBin")
listCoefsCVBinREYObs2 <- kfoldCVModelCount(ooo, dataList, "modelCountDetectBinREYObs2")
listCoefsCVBetaBinREY <- kfoldCVModelCount(ooo, dataList, "modelCountDetectBetaBinREY")

## The list
llcBin <- LLCount(dataList, listCoefsCVBinREY, ooo)
llcBinREYObs2 <- LLCount(dataList, listCoefsCVBinREY, ooo)
llcBetaBinREY <- LLCount(dataList, listCoefsCVBinREY, ooo)
```

We then carry out the same approach with the model `modelCountDetectBin`, considering yearly periods instead of two-years periods. We need to prepare the data again to test this model:

```
dataList2 <- dataCount2jags(lekcounts$lek, lekcounts$year,
                            lekcounts$nbobs, lekcounts$nbmales,
                            lekcounts$gr, as.numeric(factor(lekcounts$type)),
                            lekcounts$natun, lekcounts$year)
```

And we carry out the same approach for this model:

```
listCoefsCVBinPerYear <- kfoldCVModelCount(ooo, dataList2, "modelCountDetectBin")
llcBinPerYear <- LLCount(dataList2, listCoefsCVBinPerYear, ooo)
```

We derive the contribution of each lek to the elpd of each model:

```
elpdBin <- elpdLeks(llcBin)
elpdBinREYObs2 <- elpdLeks(llcBinREYObs2)
elpdBetaBinREY <- elpdLeks(llcBetaBinREY)
elpdBinPerYear <- elpdLeks(llcBinPerYear)
```

And we also calculate the contributions to the "corrected" elpd for each model:

```
## Number of censuses for each lek
cidl <- tapply(dataList$repetition, dataList$lek, sum)
cidl2 <- tapply(dataList2$repetition, dataList2$lek, sum)

## corrected contribution
elpdcBinREY <- elpdBinREY/cidl
elpdcBin <- elpdBin/cidl
elpdcBinREYObs2 <- elpdBinREYObs2/cidl
elpdcBetaBinREY <- elpdBetaBinREY/cidl
elpdcBinPerYear <- elpdBinPerYear/cidl
```

Finally, we calculated the elpd/elpdc, the standard deviation for each model of their difference with the best model, and we performed the randomization test for each criterion:

```
## ELPD
## Randomization test
library(ade4)
pv <- sapply(list(elpdBin, elpdBinREY, elpdBinREYObs2, elpdBetaBinREY,
            elpdBinPerYear), function(x) {

    sim <- sapply(1:1000, function(i)
        sum(sample(c(-1,1), length(elpdBinREY),
                   replace=TRUE)*(elpdBinREY-x)))
    obs <- sum(elpdBinREY-x)
    as.randtest(sim,obs)$pvalue
})

## criterion elpd
elpds <- c(sum(elpdBin),sum(elpdBinREY),
                  sum(elpdBinREYObs2),sum(elpdBetaBinREY),
           sum(elpdBinPerYear))

## SD difference with the best model
elpdDiffsd <- sqrt(length(elpdBin))*c(sd(elpdBin-elpdBinREY),
                                      0, sd(elpdBinREYObs2-elpdBinREY),
                                      sd(elpdBetaBinREY-elpdBinREY),
                                      sd(elpdBinPerYear-elpdBinREY))


## "Corrected" ELPD
## Randomization test
pvc <- sapply(list(elpdcBin, elpdcBinREY, elpdcBinREYObs2, elpdcBetaBinREY,
                   elpdcBinPerYear), function(x) {

    sim <- sapply(1:1000, function(i)
        sum(sample(c(-1,1), length(elpdBinREY),
                   replace=TRUE)*(elpdcBinREY-x)))
    obs <- sum(elpdcBinREY-x)
```

```
    as.randtest(sim,obs)$pvalue
})

## criterion
elpdcs <- c(sum(elpdcBin),sum(elpdcBinREY),
            sum(elpdcBinREYObs2),sum(elpdcBetaBinREY),
            sum(elpdcBinPerYear))

## SD Difference with the best model
elpdcDiffsd <- sqrt(length(elpdcBin))*c(sd(elpdcBin-elpdcBinREY),
                                        0, sd(elpdcBinREYObs2-elpdcBinREY),
                                        sd(elpdcBetaBinREY-elpdcBinREY),
                                        sd(elpdcBinPerYear-elpdcBinREY))


## Data.frame containing the result for elpd
dfelpd <- data.frame(Model=c("modelCountDetectBin","modelCountDetectBinREY",
                             "modelCountDetectBinREYObs2",
                             "modelCountDetectBetaBinREY",
                             "modelCountDetectBin per year"),
                     elpd=elpds,
                     SDDiff=elpdDiffsd,
                     Pvalue=round(pv,3))


## for elpdc
dfelpdc <- data.frame(Model=c("modelCountDetectBin","modelCountDetectBinREY",
                              "modelCountDetectBinREYObs2",
                              "modelCountDetectBetaBinREY",
                              "modelCountDetectBin per year"),
                      elpdc=elpdcs,
                      SDDiffc=elpdcDiffsd,
                      Pvalue=round(pvc,3))
```

3.5 Discussion

We now interpret the results of the model comparison. We first interpret the "classical" elpd:

```
dfelpd
```

```
##                           Model      elpd    SDDiff Pvalue
## 1          modelCountDetectBin -2305.641 2.8502815  0.003
## 2       modelCountDetectBinREY -2297.861 0.0000000  1.000
## 3   modelCountDetectBinREYObs2 -2299.556 2.4552215  0.267
## 4   modelCountDetectBetaBinREY -2299.329 0.9985594  0.065
## 5 modelCountDetectBin per year -2315.347 3.7525104  0.001
```

Here, for each model, we present the estimated elpd (the closer to 0 = the better), the standard deviation of the difference between the elpd of each model and the elpd of the best model (therefore equal to 0 for the best model), and the proportion of simulated differences greater than or equal to the observed difference between the elpd of a model and the elpd of the best model.

The model used in the paper (binomial distribution with the logit of the detection probability modeled as a linear effect of the number of observers + a year random effect) is the best model in the set according to

the elpd. Note that adding a quadratic effect of the number of observers or supposing an extra-binomial variation in the detection probability did not lead to a significant improvement of the prediction. However, removing the random effects of the years from the detection model led to a decrease of the elpd. Moreover, estimating one number of males in each region using one year-periods led to a strong decrease of the elpd. Clearly, our choice to consider two-year periods is a better one.

The same conclusions are reached when working on the "corrected" elpd:

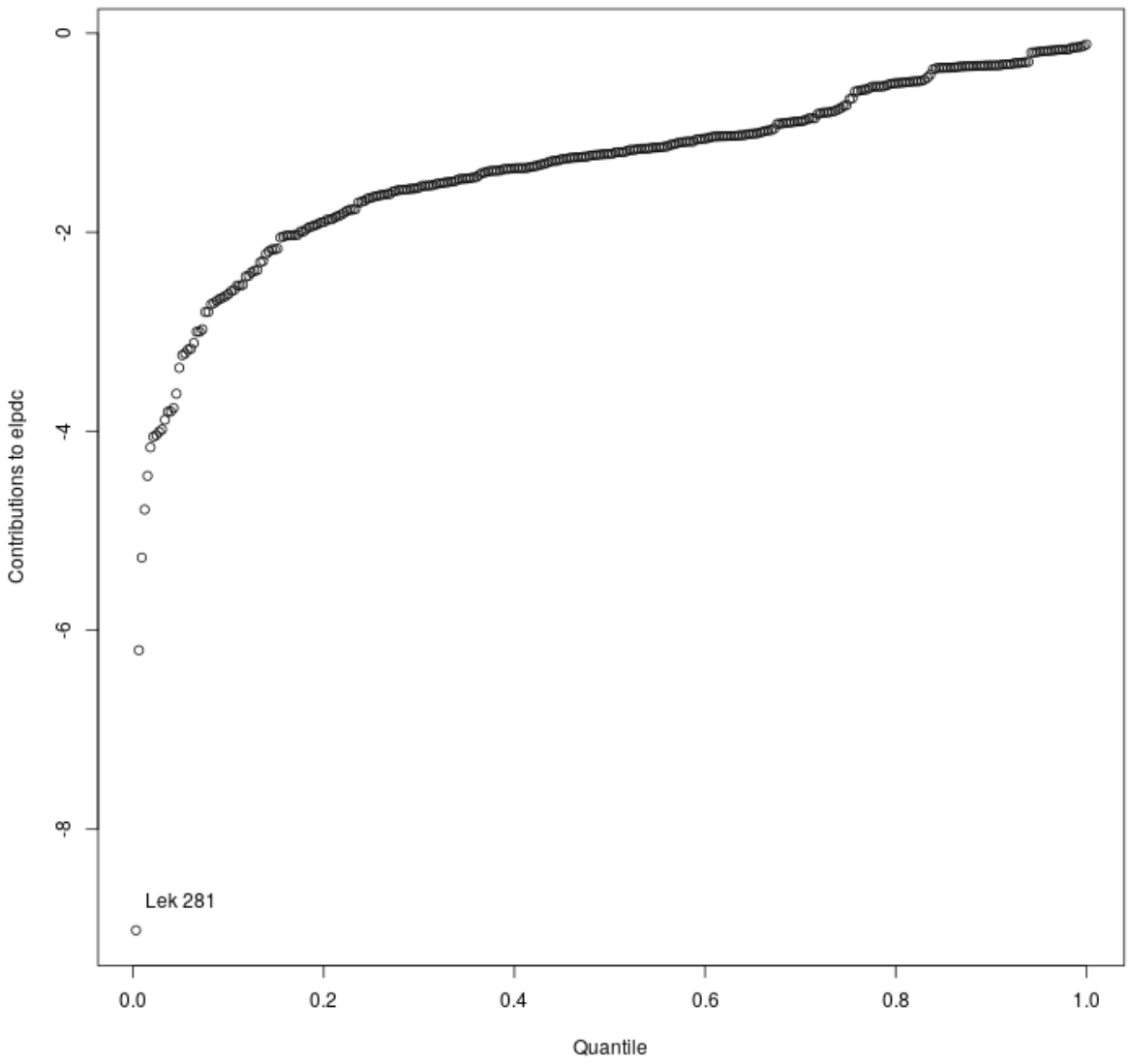
```
dfelpdc

##                            Model     elpdc   SDDiffc Pvalue
## 1          modelCountDetectBin -453.1691 0.6705215  0.011
## 2       modelCountDetectBinREY -451.5471 0.0000000  1.000
## 3   modelCountDetectBinREYObs2 -451.8273 0.5481937  0.295
## 4   modelCountDetectBetaBinREY -451.6960 0.2046681  0.276
## 5 modelCountDetectBin per year -455.4702 1.0848683  0.001
```

4 The lek # 281

We have seen in the section 2.1.4 that the lek # 281 was characterized by a very large residual in our model. This lek is a UL discovered by the grid cell searches carried out in 2013, and it is characterized by a very large number of detected animals (the two censuses carried out on this lek resulted in 18 and 17 males respectively).

The examination of the quantile plot of the contributions of each lek to the corrected elpd shows that the lek 281 contributes significantly to the elpd:

```
plot(c(1:length(elpdcBinREY))/length(elpdcBinREY),
     sort(elpdcBinREY), xlab="Quantile",
     ylab="Contributions to elpdc")
text(0.05, -8.71, "Lek 281")
```



Other state models that we have tried during previous years on this dataset identify the same problem: this lek is an outlier characterized by a much larger number of detected males than other ULs. Since the number of detected ULs is low (32 leks), any outlier will have a strong effect on the inference.

We wanted to assess more precisely the effect of this lek on our inference, by trying to fit the model again without this lek, and by estimating again the number of males in each region. WARNING: THIS CALCULATION TAKES A VERY LONG TIME (several hours)!!!! Fortunately, the package contains the result `coefModelm281` of the fit as a dataset in the package:

```
lcb <- lekcounts %>% filter(lek!=281)
lcb$lek <- as.numeric(factor(lcb$lek))
dataListwo281 <- dataCount2jags(lcb$lek, lcb$period,
                                lcb$nbobs, lcb$nbmales,
                                lcb$gr, as.numeric(factor(lcb$type)),
                                lcb$natun, lcb$year)

coefModelm281 <- fitModelCount(dataListwo281, "modelCountDetectBinREY")
```

We then estimate the number of males in each region with this new model:

```
nmwo281 <- estimateNmales(coefModelm281, coefModelULPresence,
                          gridFrame, NKAL, NKIL)
summary(nmwo281)

## Estimates of the number of males during the 5 periods
## in the whole Pyrenees Mountains  (median and 80% CI):
##
##   Point.est    CIlow     CIup
## 1  1782.205 1488.828 2222.923
## 2  1795.783 1489.977 2243.659
## 3  1803.610 1505.648 2246.265
## 4  1769.254 1474.038 2189.760
## 5  1637.393 1367.850 2013.980
##
## Change rate between period 1 and period 5 is: -7.39% (80% CI: -17.59% -- 0.52%)
##
## Probability of scenarii (decrease:< -10%; stability:-10--10%; increase:>10%):
##
##    scenario probability
## 1  decrease      0.3620
## 2 stability      0.6355
## 3  increase      0.0025
```

We recall the estimates of our model:

```
summary(nm)

## Estimates of the number of males during the 5 periods
## in the whole Pyrenees Mountains  (median and 80% CI):
##
##   Point.est    CIlow     CIup
## 1  1871.789 1542.350 2350.656
## 2  1922.285 1570.448 2440.920
## 3  1901.102 1562.484 2387.119
## 4  1841.119 1516.075 2308.645
## 5  1694.132 1398.089 2129.504
##
## Change rate between period 1 and period 5 is: -8.49% (80% CI: -20.24% -- 0.13%)
##
## Probability of scenarii (decrease:< -10%; stability:-10--10%; increase:>10%):
##
##    scenario probability
## 1  decrease     0.43100
## 2 stability     0.56575
## 3  increase     0.00325
```

Removing this lek from the dataset results in a decrease of the estimated population size of about 70 to 100 males, i.e. about a 5% decrease in the population size estimated for the whole mountain range. Note that the lek 281 was located in the geographical region 1 and discovered in period 2, so that we could expect a larger effect of this lek in this region during this period. We compare the estimates of region 1 with this lek...

```
summary(nm,1)

## Estimates of the number of males during the 5 periods
## in the geographic region 1  (median and 80% CI):
```

```
##
##   Point.est     CIlow     CIup
## 1  155.9515 111.19115 227.6323
## 2  154.9948 109.65762 219.2993
## 3  149.0112 106.98677 213.7739
## 4  145.4841 104.07529 209.1013
## 5  135.4594  95.31548 198.5411
##
## Change rate between period 1 and period 5 is: -12.1% (80% CI: -28.59% -- 2.13%)
##
## Probability of scenarii (decrease:< -10%; stability:-10--10%; increase:>10%):
##
##    scenario probability
## 1  decrease     0.57125
## 2 stability     0.40050
## 3  increase     0.02825
```

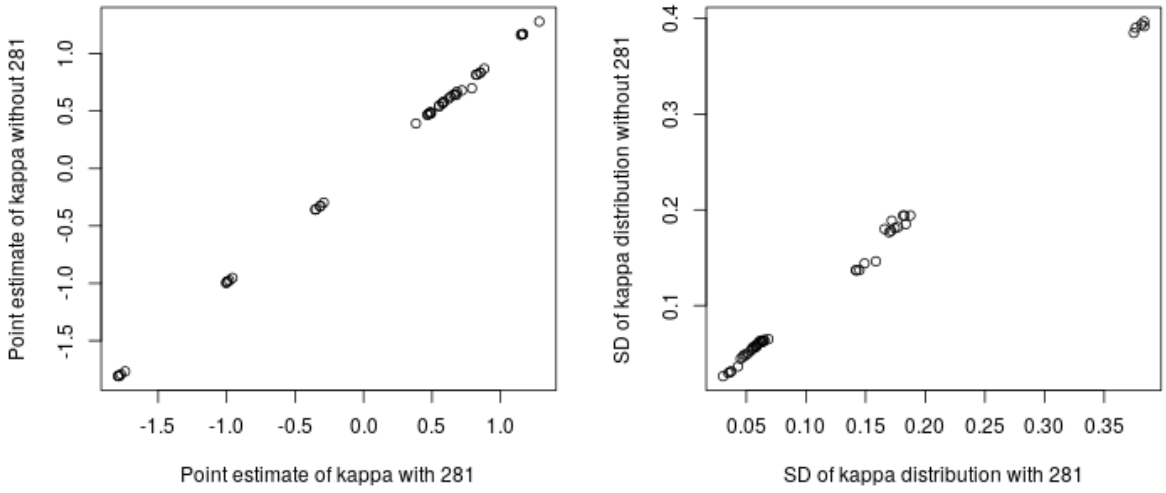
...and without this lek:

```
summary(nmwo281,1)


## Estimates of the number of males during the 5 periods
## in the geographic region 1  (median and 80% CI):
##
##   Point.est     CIlow     CIup
## 1  149.1638 105.41717 211.2579
## 2  143.5529 102.32480 200.2430
## 3  142.5922 102.24431 198.9411
## 4  140.8573  99.66753 197.3924
## 5  131.0549  92.35818 188.5257
##
## Change rate between period 1 and period 5 is: -10.53% (80% CI: -26.86% -- 3.22%)
##
## Probability of scenarii (decrease:< -10%; stability:-10--10%; increase:>10%):
##
##    scenario probability
## 1  decrease     0.51825
## 2 stability     0.45150
## 3  increase     0.03025
```

The removal of this lek from the dataset results in a decrease of the estimated population size in this region of about 10 males, i.e. about a 6.5% decrease in the population size. Actually, this lek has a strong effect on the estimation of the number of males, has the same effect in all regions. Actually, the estimates of the random effects $\kappa_{g(i),\ell(i),b(t)}$, which represent how the median number of males on a lek varies across regions and periods, are very similar whether we remove lek 281 or not:

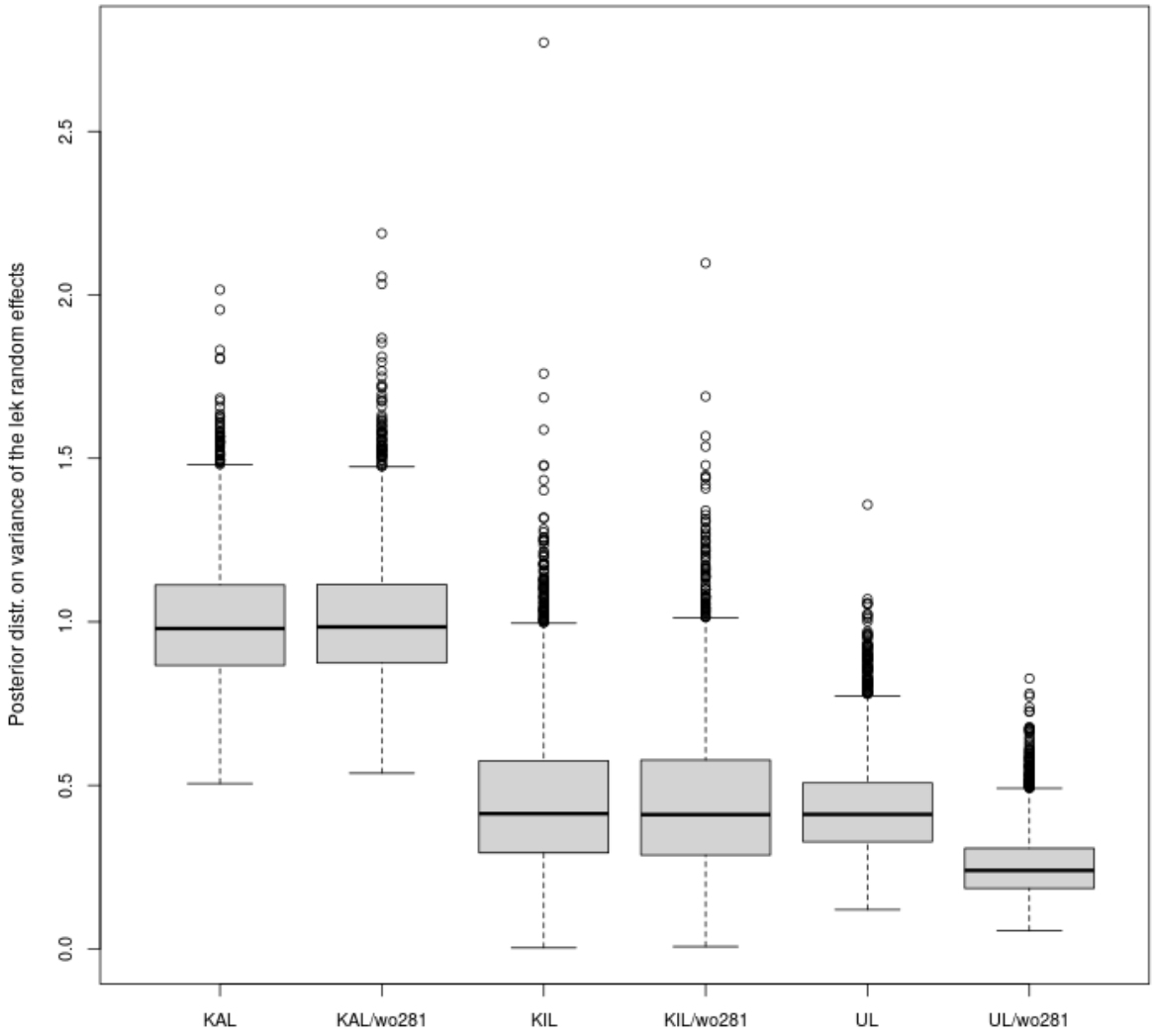
```
library(matrixStats)
kap <- MCMCchains(coefModelm281, "kappa")
kapb <- MCMCchains(coefModelCountDetectBinREY, "kappa")
par(mfrow=c(1,2))
cmbk <- colMeans(kapb)
cmbv <- colVars(kapb)
cmk <- colMeans(kap)
cmv <- colVars(kap)

## Remove the "virtual levels" included in the model to have
```

```
## a square matrix region x period x types of leks
## (we included these levels because we do not have the
## same number of regions for all types of lek)
cmbk <- cmbk[cmv<8]
cmbv <- cmbv[cmv<8]
cmk <- cmk[cmv<8]
cmv <- cmv[cmv<8]


plot(cmbk, cmk, xlab="Point estimate of kappa with 281",
     ylab="Point estimate of kappa without 281")
plot(cmbv, cmv, xlab="SD of kappa distribution with 281",
     ylab="SD of kappa distribution without 281")
```



Actually, this lek increases the estimate of the variance of the lek random effects:

```
set <- 1/MCMCchains(coefModelm281, "sigmaeta")
setb <- 1/MCMCchains(coefModelCountDetectBinREY, "sigmaeta")
colnames(setb) <- c("KAL","KIL","UL")
colnames(set) <- paste0(c("KAL","KIL","UL"),"/wo281")
setg <- cbind(setb,set)[,c(1,4,2,5,3,6)]
boxplot(setg, cex.axis=0.9, ylab="Posterior distr. on variance of the lek random effects")
```



The estimated variance for ULs is much smaller when the lek 281 is removed. Examination of other parameters show that adding/removing census data of the lek # 281 does not change the posterior distribution.

Removing this lek from our dataset would lead to a decrease of about 5% of our population size estimate. We have no reason to exclude this lek, as it has been discovered by our protocol, and actually "represents" other ULs of similar size, which – although they are probably not very frequent in the mountain range – do exist but are not yet discovered. Today, as the number of ULs detected during grid cells searches is small, any big lek will have a strong effect on the inference, which is what we show here. Today, this lek is an outlier, and as such, has a significant effect on our inference, though not that strong (probably less than 5%). But as more and more data on ULs will be collected in the future, the weight of this lek on our estimate will decrease and we will have a finer understanding of the distribution of the number of males on ULs.

5 Sensitivity of our model to the violation of several hypotheses

5.1 Criticisms of [LIN18] and [BAR17]

Two recent papers by [LIN18] and [BAR17] identified many problems with N-mixture models that can strongly affect the population size estimation by these models. In this section, we review the criticisms by these authors about N-mixture models to consider whether they apply to our model. These authors consider the inference of abundance N drawn from a Poisson distribution and a binomial detection process characterized by a detection probability p.

First, [BAR17] suggests that in some studies, the abundance N may not exist as a parameter. We understand that this situation may occur in some studies, e.g. in some point count studies where the area delimited by a "site" may not be clearly defined. However, in our study, we suppose that each capercaillie cock of the mountain range is located on only one lek. Therefore, the set of leks can be considered as a contraction of the space used by capercaillie cocks, and the abundance does exist as a parameter for each lek.

Another issue arises when the detection probability $p \to 0$; in this case the binomial distribution describing the detection process converges towards a Poisson distribution with parameter $\lambda = Np$. In other words, the parameters N and p are not separable when p is small, making the inference on N alone difficult. However, in our study, the cock detection probability was estimated to be of ≈ 0.6, so that this issue is not a concern in our study.

Actually, the most serious criticisms by these authors concern the effect of unaccounted variation in detection probability. These authors show, in the case of a state model corresponding to a simple Poisson distribution that even a small amount of unaccounted variation can lead to a considerable bias in the estimation of N. They show the same problem with accidental double counting. Thus, [LIN18] note: "*2% rate of double counts or a standard deviation of 2% in detection rates might seem insubstantial, but each is large enough to produce >20% bias in estimation of mean abundance*". This is a worrying result, as no model could ascertain that the amount of unaccounted variation in the detection process is so small. In the next two sections, we carry out some simulations to test the effect of unaccounted variation in detection probability and accidental double counting on the estimation of the number of males.

5.2 Unaccounted variation in detection probability

5.2.1 Simulation of both the state and detection process

We tested the effect of unaccounted variation in detection probability on the estimation of the number of males in the whole mountain range. We used the model fitted in section 2.1.2 to generate new datasets (i.e. simulating the state process, and then the detection process), adding a level of unaccounted variation in the detection probability. More precisely, we used the same state model as the one used in the paper to generate numbers of males on the sampled leks. We also used the same model to simulate the *average* probability of detection for each census. For the census k of year t on lek i, this average probability is $d_{i,t,k}$ and the actual probability is supposed to be drawn from a beta distribution:

$$p_{i,t,k} \sim \text{Beta}\left(d_{i,t,k} \times \frac{1-\delta^2}{\delta^2}, (1-d_{i,t,k}) \times \frac{1-\delta^2}{\delta^2}\right)$$

The parameter δ^2 controls the amount of extra-binomial variation added to the detection process. We considered the followwing values for the parameter δ^2:

```
delta2b <- c(0, 0.001, 0.005, 0.01, 0.05, 0.1, 0.3)
```

For each value of δ^2, we simulated 10 datasets, and we estimated the posterior distribution of the number of males. We used the function `simulateDataList`, included in the package to simulate this data

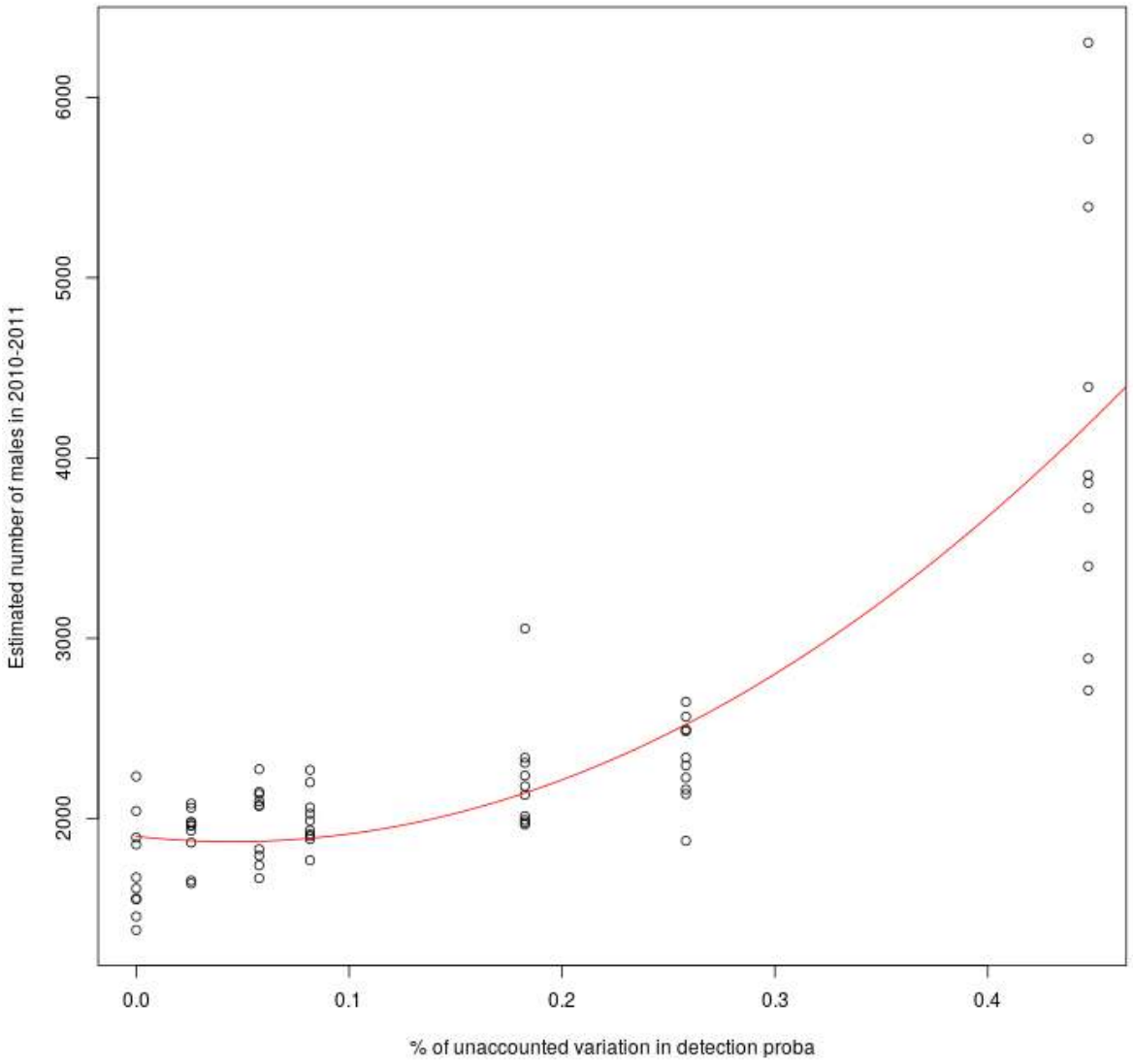
generating process. We then calculated the median of this distribution. WARNING!!! THIS CALCULATION TAKES A VERY LONG TIME (several hours)!!!! Fortunately, the package contains the result `medianNmalesBB` of this calculation (matrix with 10 columns – the 10 repetitions – and 7 rows – the 7 values of δ^2 – containing the median of the posterior distributions):

```
libtt <- list()

for (j in 1:10) {
    cat("############################",
        "\n############################",
        "\n############################",
        "\n############################",
        "\n### Iteration", j,"\n\n\n")
    liresuBeta <- list()
    for (i in 1:7) {
        cat("############################\n### Model", i,"\n\n\n")
        if (i!=1) {
            sdl <- simulateDataList(coefModelCountDetectBinREY,
                                    dataList, betaBinDelta=delta2b[i])
        } else {
            sdl <- simulateDataList(coefModelCountDetectBinREY, dataList)
        }
        fm <- fitModelCount(sdl, "modelCountDetectBinREY", n.iter=30000, thin=30)
        liresuBeta[[i]] <- list(dataList=sdl, coefs=fm)
        saveRDS(liresuBeta, file="liresuBeta.Rds")
    }
    libtt[[j]] <- liresuBeta
    saveRDS(libtt, file="libtt.Rds")
}


listNmalesBB <- lapply(libtt, function(liresuBeta)
    lapply(liresuBeta, function(x)
        estimateNmales(x$coefs, coefModelULPresence, gridFrame, NKAL, NKIL)))

medianNmalesBB <- sapply(listNmalesBB, function(z)
    sapply(z, function(x) median(getTotal(x)[,1])))
```

We present below how the median of the estimated posterior distribution on the number of males for the reference period 2010-2011 in the whole mountain range varies with the proportion of unaccounted variation. As the estimated detection probability in our study is about 0.6, we calculated the coefficient of variation of a beta distribution of the average detection probabilities was equal to 0.6, for each value of δ^2. This allowed to display the relationship between the proportion of unaccounted variation and the estimated number of males:

```
## coefficient of variation
coefvar <- c(0,sapply(delta2b[-1], function(delta2) {
    alpha <- 0.6*(1-delta2)/delta2
    beta <- 0.4*(1-delta2)/delta2
    sqrt(alpha*beta/((alpha+beta)^2 * (alpha+beta+1)))/0.6
}))


plot(rep(coefvar, 10),as.vector(medianNmalesBB),
     xlab="% of unaccounted variation in detection proba",
     ylab="Estimated number of males in 2010-2011")
xy <- data.frame(x=rep(coefvar, 10), y=as.vector(medianNmalesBB))
```

```
xv <- seq(0,0.5,length=200)
lines(xv, predict(lm(y~x+I(x^2), data=xy), newdata=data.frame(x=xv)), col="red")
```



The case $\delta^2 = 0$ corresponds to zero unaccounted extra-binomial variation. Note that the overestimation in the number of males is not linearly related to the coefficient of variation of the detection probability.

Although increasing the amount of unaccounted variation in the detection probability indeed results in an increase in the estimated number of males, our model seems quite robust to this source of bias: even with $\sim 10\%$ of unexplained variability in the detection probability, the resulting bias is nearly unnoticeable in comparison to the imprecision of our estimation.

5.2.2 Simulating the detection process only

In the previous section, we have assessed the effect of unaccounted heterogeneity in the detection process on the estimation of the number of males by simulating the whole process (both the state process generating the true number of males on sampled leks, and the detection process), while adding a certain amount

of unaccounted variability in the process. However, since we are working on unaccounted variation in detection probability, it would be interesting to keep the true number of males fixed in our simulations while varying the amount of unaccounted variation in detection probability.

We used the function `simulateN` to simulate the true number of males present on the sampled leks/periods.

```
set.seed(1)
simn <- simulateN(coefModelCountDetectBinREY,dataList)
```

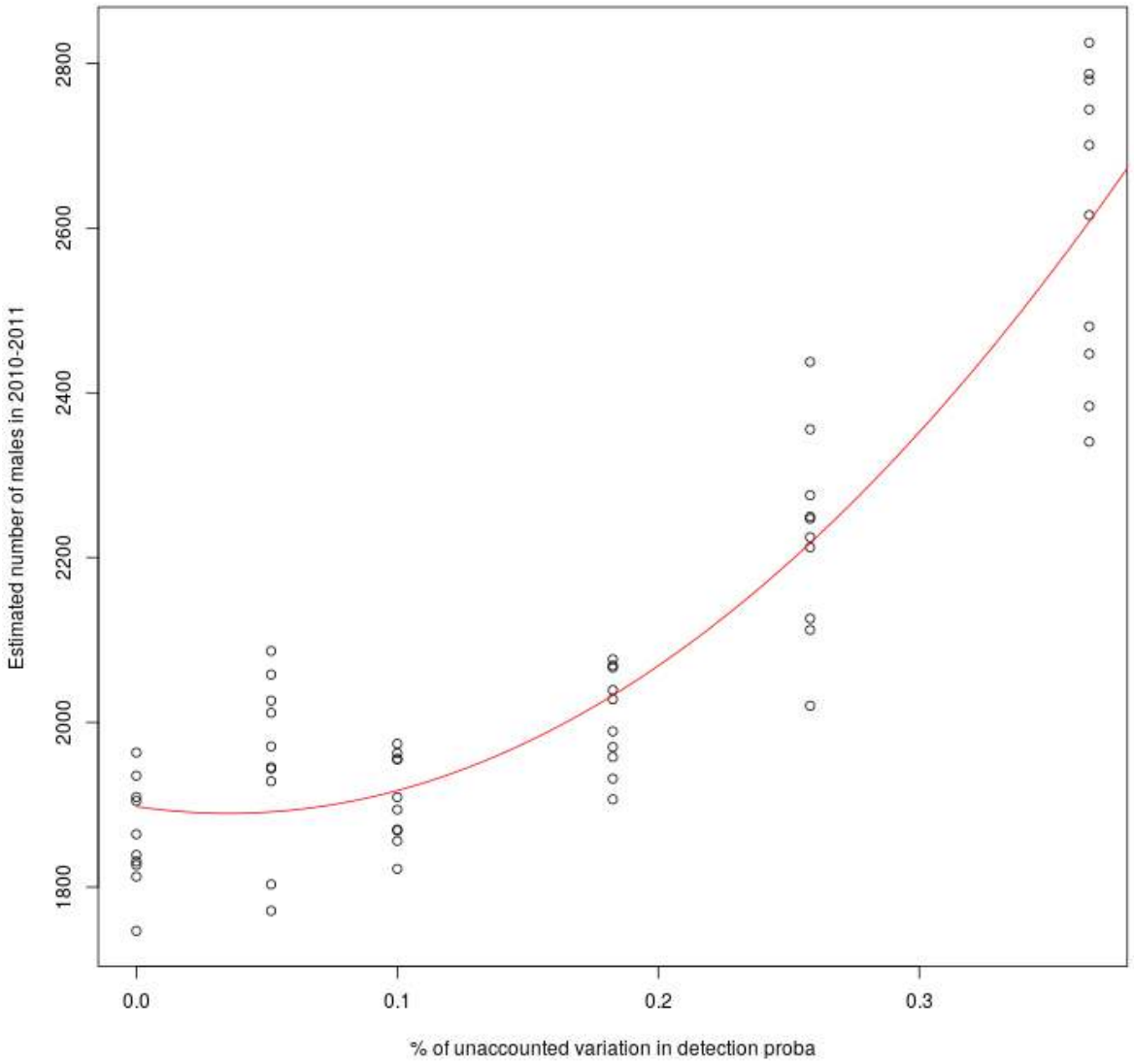
Then, for this particular simulated true number of males, we simulated the detection process with various levels of unaccounted detection probability. WARNING!!! THIS CALCULATION TAKES A VERY LONG TIME (several hours)!!!! Fortunately, the package contains the result `medianNmalesBB2` of this calculation (matrix with 10 columns – the 10 repetitions – and 5 rows – the 5 values of h – containing the median of the posterior distributions):

```
delta2b <- c(0, 0.004, 0.015, 0.05, 0.1, 0.2)
libtt2 <- list()
for (j in 1:10) {
    cat("############################",
        "\n############################",
        "\n############################",
        "\n############################",
        "\n### Iteration", j,"\n\n\n")
    liresuBeta <- list()
    for (i in 1:6) {
        cat("############################\n### Model", i,"\n\n\n")
        if (i!=1) {
            sdl <- simulateDataList2(coefModelCountDetectBinREY,dataList,
                                     simn, betaBinDelta=delta2b[i])
        } else {
            sdl <- simulateDataList2(coefModelCountDetectBinREY,
                                     dataList, simn)
        }
        fm <- fitModelCount(sdl, "modelCountDetectBinREY", n.iter=10000, thin=10)
        liresuBeta[[i]] <- list(dataList=sdl, coefs=fm)
    }
    libtt2[[j]] <- liresuBeta
    saveRDS(libtt2, file="libtt2.Rds")
}


listNmalesBB2 <- lapply(libtt2, function(liresuBeta)
    lapply(liresuBeta, function(x)
        estimateNmales(x$coefs, coefModelULPresence,
                       gridFrame, NKAL, NKIL)))

medianNmalesBB2 <- sapply(listNmalesBB2, function(z)
    sapply(z, function(x) mean(getTotal(x)[,1])))
save(medianNmalesBB2, file="medianNmalesBB2.rda")
```

We show below how the median of the estimated posterior distribution on the number of males for the reference period 2010-2011 in the whole mountain range varies with the coefficient of variation of the detection probability when this probability is equal to 0.6:

```
delta2b <- c(0, 0.004, 0.015, 0.05, 0.1, 0.2)
```

```
## coefficient of variation
coefvar <- c(0,sapply(delta2b[-1], function(delta2) {
    alpha <- 0.6*(1-delta2)/delta2
    beta <- 0.4*(1-delta2)/delta2
    sqrt(alpha*beta/((alpha+beta)^2 * (alpha+beta+1)))/0.6
}))


plot(rep(coefvar, 10),as.vector(medianNmalesBB2),
     xlab="% of unaccounted variation in detection proba",
     ylab="Estimated number of males in 2010-2011")
xy <- data.frame(x=rep(coefvar, 10), y=as.vector(medianNmalesBB2))
xv <- seq(0,0.5,length=200)
lines(xv, predict(lm(y~x+I(x^2), data=xy), newdata=data.frame(x=xv)), col="red")
```



Keeping fixed the actual number of males while varying the unaccounted variation in detection probability does not change the results obtained in the previous section. The effect of unaccounted variation in detection probability is unnoticeable when the coefficient of variation of this probability is <15%. When

this coefficient of variation reaches 25%, this results in a point estimate of $\sim$ 2200 males instead of $\sim$ 1900, i.e. (2200-1900)/1900 $\approx$ 15% overestimate.

This result is very different from the 20% bias in estimated number of males resulting from the 2% unaccounted variation in [LIN18]. Actually, these authors considered a very simple Poisson state model: with their simple model, any unaccounted heterogeneity in detection probability will result in an increase of the estimated Poisson parameter. Our model is more complex, with many random effects accounting for the hierarchical structure of the state process: the unaccounted heterogeneity in detection probability is likely absorbed by the various random effects of the state model, which limits this effect.

In conclusion, the unaccounted heterogeneity in detection process is unlikely to strongly affect our inference.

5.3 Effect of accidental double counting

5.3.1 Simulation of both the state and detection process

We also tested the effect of accidental double counting on our estimation. As for the assessment of the effect of unaccounted heterogeneity, we first generated datasets by simulating the state process, and then the detection process. We also used the model fitted in section 2.1.2 to generate new datasets, but this time, we added a random proportion of animals counted twice to the simulated censuses. More precisely, we used the same state model as the one used in the paper to generate numbers of males on the sampled leks. We also used the same model to simulate the probability of detection for each census. If $y_{i,t,k}$ is the number of distinct males detected on lek i during census k of year t, then we suppose that the number of animals counted twice during this census is randomly drawn from a Binomial distribution $\mathcal{B}(y_{i,t,k}, h)$, where h is the proportion of animals counted twice. We tested the following values of h:

```
doubleCountspb <- c(0.05, 0.1, 0.2, 0.3, 0.5)
```

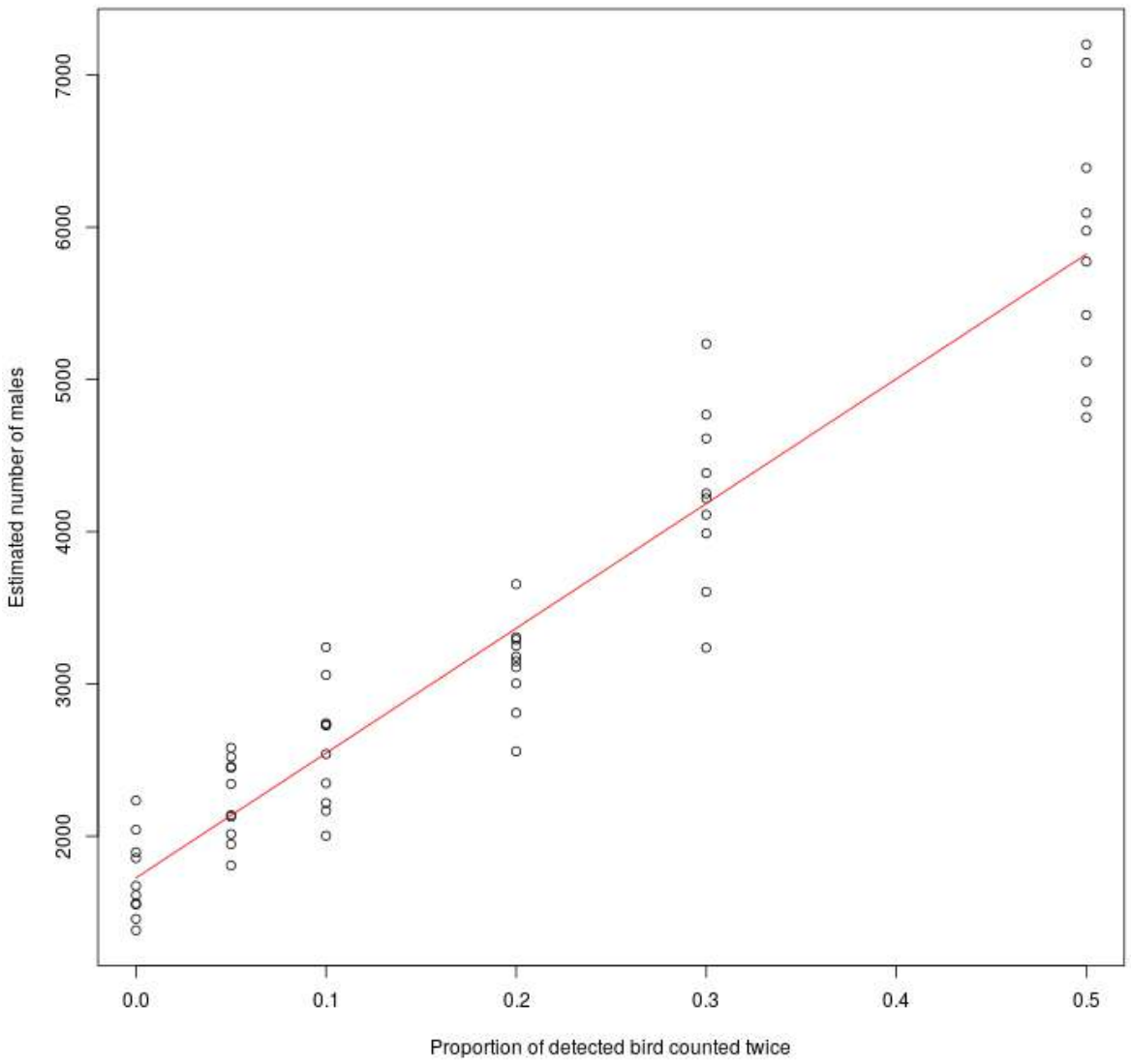
For each value of h, we simulated 10 datasets, and we estimated the posterior distribution of the number of males. We again used the function `simulateDataList`, included in the package to simulate this data generating process. We then calculated the median of this distribution. WARNING!!! THIS CALCULATION TAKES A VERY LONG TIME (several hours)!!!! Fortunately, the package contains the result `medianNmalesBB` of this calculation (matrix with 10 columns – the 10 repetitions – and 5 rows – the 5 values of h – containing the median of the posterior distributions):

```
lidtt <- list()
for (j in 1:10) {
    cat("############################",
        "\n############################",
        "\n############################",
        "\n############################",
        "\n### Iteration", j,"\n\n\n")
    liresudc <- list()
    for (i in 1:5) {
        cat("############################\n### Model", i,"\n\n\n")
        sdl <- simulateDataList(coefModelCountDetectBinREY, dataList,
                                doubleCountsp=doubleCountspb[i])
        fm <- fitModelCount(sdl, "modelCountDetectBinREY", n.iter=30000, thin=30)
        liresudc[[i]] <- list(dataList=sdl, coefs=fm)
        saveRDS(liresudc, file="liresudc.Rds")
    }
    lidtt[[j]] <- liresudc
    saveRDS(lidtt, file="lidtt.Rds")
}
```

```
listNmalesDC <- lapply(lidtt, function(liresudc)
    lapply(liresudc, function(x)
        estimateNmales(x$coefs, coefModelULPresence, gridFrame, NKAL, NKIL)))

medianNmalesDC <- sapply(listNmalesDC,
                         function(z)
    sapply(z, function(x) median(getTotal(x)[,1])))
```

We present below how the median of the estimated posterior distribution on the number of males for the reference period 2010-2011 in the whole mountain range varies with the proportion of accidental double counting:

```
## Add the case where h=0 (simulated in the previous section)
plot(rep(c(0,doubleCountspb), 10),
     as.vector(rbind(medianNmalesBB[1,], medianNmalesDC)),
     xlab="Proportion of detected bird counted twice",
     ylab="Estimated number of males")
xy <- data.frame(x=rep(c(0,doubleCountspb), 10),
                 y=as.vector(rbind(medianNmalesBB[1,], medianNmalesDC)))
xv <- seq(0,0.5,length=200)
lines(xv, predict(lm(y~x, data=xy), newdata=data.frame(x=xv)), col="red")
```



Accidental double counting has a much stronger effect on the estimation than unaccounted variation in detection probability. Indeed, even a moderate average proportion of 10% of detected animals counted twice results in an estimate of about 2550 animals (instead of the average 1900 obtained in this set of simulations). This corresponds to an overestimation of $\sim 34\%$. We therefore confirm the results of [LIN18] here.

5.3.2 Simulating the detection process only

In the previous section, we have assessed the effect of accidental double counting on the estimation of the number of males by simulating the whole process (both the state process generating the true number of males on sampled leks, and the detection process), while adding a certain number of males counted twice. However, again, we are working on the detection process so that it would be interesting to keep the true number of males fixed in our simulations while varying the proportion of detected males counted twice.

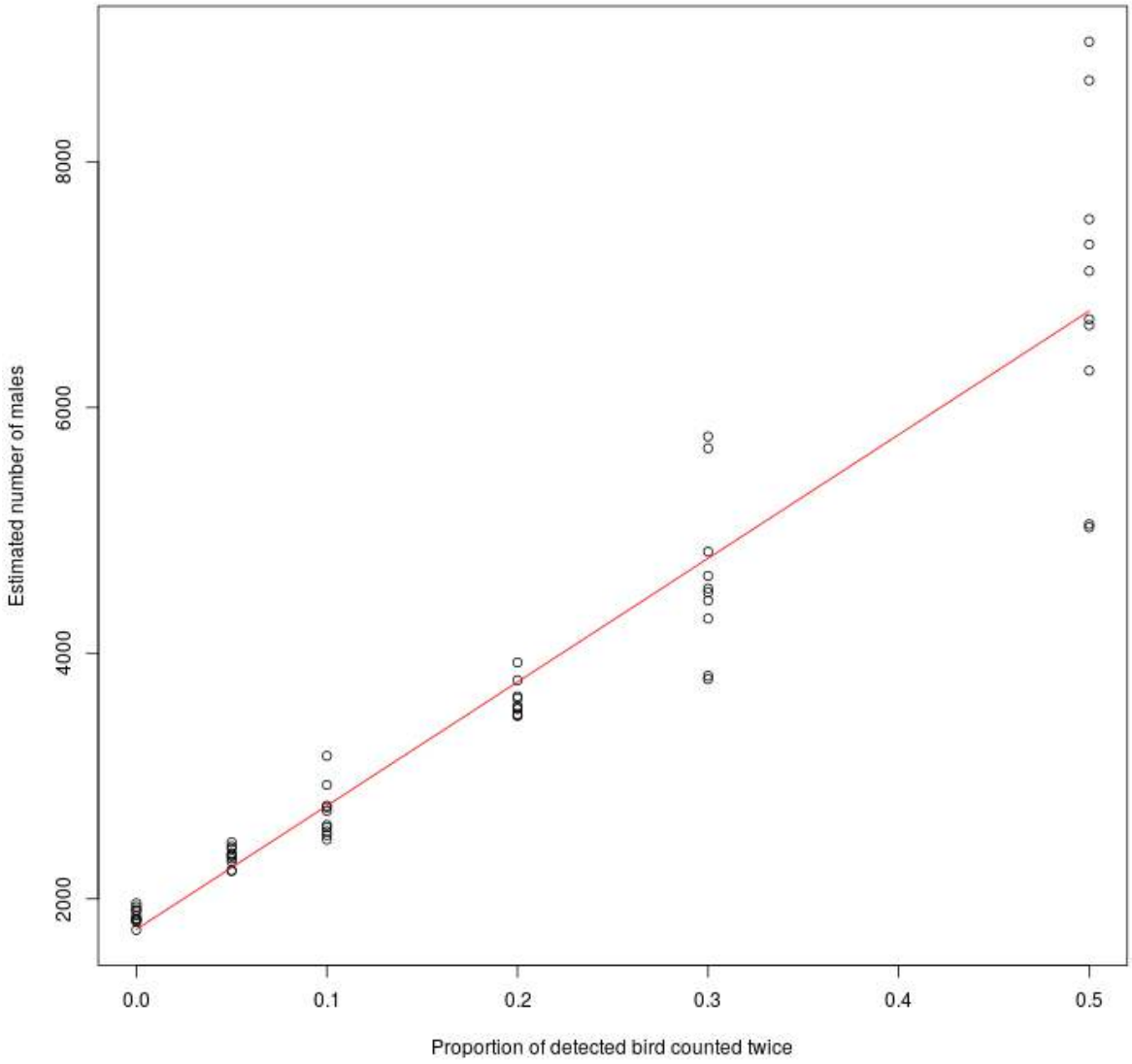
We used the same vector `simn` of true number of males simulated in section 5.2.2 as the "true" state of the system, and for this particular simulated true number of males, we simulated the detection process with various levels of accidental double counting. WARNING!!! THIS CALCULATION TAKES A VERY

LONG TIME (several hours)!!!! Fortunately, the package contains the result `medianNmalesDC2` of this calculation (matrix with 10 columns – the 10 repetitions – and 5 rows – the 5 values of h – containing the median of the posterior distributions):

```
doubleCountspb <- c(0.05, 0.1, 0.2, 0.3, 0.5)

lidtt2 <- list()
for (j in 1:10) {
    cat("############################",
        "\n############################",
        "\n############################",
        "\n############################",
        "\n### Iteration", j,"\n\n\n")
    liresudc <- list()
    for (i in 1:5) {
        cat("############################\n### Model", i,"\n\n\n")
        sdl <- simulateDataList2(coefModelCountDetectBinREY,dataList, simn,
                                 doubleCountsp=doubleCountspb[i])
        fm <- fitModelCount(sdl, "modelCountDetectBinREY", n.iter=10000, thin=10)
        liresudc[[i]] <- list(dataList=sdl, coefs=fm)
        saveRDS(liresudc, file="liresudc.Rds")
    }
    lidtt2[[j]] <- liresudc
    saveRDS(lidtt2, file="lidtt2.Rds")
}


listNmalesDC2 <- lapply(lidtt2, function(liresuBeta)
    lapply(liresuBeta, function(x)
        estimateNmales(x$coefs, coefModelULPresence, gridFrame, NKAL, NKIL)))

medianNmalesDC2 <- sapply(listNmalesDC2, function(z)
    sapply(z, function(x) mean(getTotal(x)[,1])))
```

We show below how the median of the estimated posterior distribution on the number of males for the reference period 2010-2011 in the whole mountain range varies with the proportion of accidental double counting:

```
doubleCountspb <- c(0.05, 0.1, 0.2, 0.3, 0.5)

## Note that we add the case where h=0 (simulated in the previous section)
plot(rep(c(0,doubleCountspb), 10),
     as.vector(rbind(medianNmalesBB2[1,], medianNmalesDC2)),
     xlab="Proportion of detected bird counted twice",
     ylab="Estimated number of males")
xy <- data.frame(x=rep(c(0,doubleCountspb), 10),
                 y=as.vector(rbind(medianNmalesBB2[1,], medianNmalesDC2)))
xv <- seq(0,0.5,length=200)
lines(xv, predict(lm(y~x, data=xy), newdata=data.frame(x=xv)), col="red")
```



Keeping fixed the actual number of males while varying the probability that a detected male was counted twice does not change the results obtained in the previous section. Even a moderate average proportion of 10% of detected animals counted twice results in an estimate of about 2750 animals (instead of the average 1900 obtained for a zero proportion). This corresponds to an overestimation of $\sim 45\%$.

In conclusion, the unaccounted double counting during censuses can have a very strong effect on our estimation.

6 History of the program

The original sampling design was developed in 2009, and we first applied it during the period 2010–2011. Our initial aim was to estimate the true number of males at different spatial scales over the mountain range, the smaller scale being the natural unit. For this reason, we initially decided to stratify the sampling of both grid cells for cell searchs and known leks for censuses by natural unit. First, a sample of natural units was drawn. The sample of KILs/KALs and grid cells were sampled in those natural units. The same sample of natural units were monitored for the first three periods, so that at the end of the third period, all the grid cells of those units had been searched for new ULs. For the fourth period,

a new sample of natural units was drawn (partially matching the previously sampled NUs, to allow a longitudinal monitoring at least for some leks).

After the first application in 2010–2011, only 6 new ULs were discovered following the searches carried out in 77 grid cells carried out this year. Thus, despite a substantial field work, the number of discovered ULs was to small to allow the fit of a statistical distribution of the true number of males present on ULs. We therefore made the assumption that the number of males on UL had a statistical distribution similar to the number of males on KILs. Indeed:

- We thought at that time that most large leks on the mountain range had already been discovered and were already registered as KALs in our sampling frame. Thus, we expected that discovered ULs were small leks that appeared recently on the mountain range, probably characterized by a tendency to increase. In other words, we expected that ULs had a behavior similar to KILs.

- The discovered ULs were indeed characterized by a small number of detected males (four leks with one detected male and one lek with three detected males), which seemed to confirm this assumption.

- One of the "discovered" ULs was actually already known by the observers before the sampling. It was a small lek that was known but which had never been censused. However the observer had forgotten to report it to the program managers prior to the definition of the sampling design in 2009. In other words, this UL shared all characteristics of a KIL.

- None of the goodness of fit tests carried out at that time seemed to indicate that we were wrong in making the assumption that ULs and KILs were behaving similarly.

However, with time, the number of ULs discovered after grid cell searchs increased, and after the period 4 (in 2017), we realized that this assumption did not hold. In particular, the true number of males was much larger on ULs than on KILs. This matched the impression of some partners of the program that this number was probably underestimated.

We therefore decided to fit a separate statistical distribution for the number of males on ULs after period 4. This had the unfortunate consequence to increase the uncertainty of our estimation. Indeed, by essence, the category of ULs is not well known, and only a small number of them is censused every year (although this number is increasing after each period). The uncertainty on the estimated number of males on all ULs results from the fact that we need to estimate both the number of ULs and the distribution of the true number of males on ULs. Since, by definition, none of these components were known at the onset of the program, the total number of males present on ULs is the most uncertain quantity in our model. Separating the ULs and KILs led to a much higher imprecision of the estimates of the total number of males, since it implied that the proportion of the population present in a badly known compartment of the population was larger than previously imagined.

As a result, we changed our sampling approach of grid cells. The only way to decrease uncertainty was to put a maximum effort in the discovery of new ULs. We then increased the effort put in grid cell searches for the period 2018–2019. Moreover, we modified the sampling design to increase the probability that a grid cell search result into the discovery of new ULs. We stopped to restrain our sampling to a small set of natural units, and sampled grid cells among all grid cells of a geographic region instead.

We used a map of the area of potential capercaillie presence defined by a group of experts in 2009, before the onset of the program as a guide to select grid cells. We sampled the grid cells to be searched with a probability proportional to the proportion of the cells covered by the area of capercaillie potential presence. Moreover, we decided to no longer stratify the grid cells by natural units: it appeared that

there were no clear differences of the probability of presence of UL between the natural units. Avoiding this stratification allowed to sample grid cells over the whole mountain range and thereby maximize the probability to find new ULs in sampled grid cells.

However, between 2010 and 2017, many ULs discovered by the network of observers outside the monitoring framework (e.g. resulting either from accidental discovery, from compilation of local knowledge, or from local initiatives leading to the search for new leks) were reported to the program managers. When a sampled grid cell contained such an UL, no search was organised for this cell, and the presence of an UL was reported. Our model differenciates these ULs from the ULs discovered following grid cells.

References

[BAR17] Barker, R. J.; Schofield, M. R.; Link, W. A. & Sauer, J. R. 2018. On the reliability of N-mixture models for count data. *Biometrics*, 74, 369-377.

[GAR00] Garrett, E.S. & Zeger, E.L. 2000. Latent class model diagnosis. *Biometrics*, 56, 1055–1067.

[LIN18] Link, W. A.; Schofield, M. R.; Barker, R. J. & Sauer, J. R. 2018. On the robustness of N-mixture models. *Ecology*, 99, 1547-1551.

[MAR11] Martin, J.; Royle, J. A.; Mackenzie, D. I.; Edwards, H. H.; Kery, M. & Gardner, B. 2011. Accounting for non-independent detection when estimating abundance of organisms with a Bayesian approach. *Methods in Ecology and Evolution*, 2, 595-601

[MER19] Merkle, E. C.; Furr, D. & Rabe-Hesketh, S. 2019. Bayesian Comparison of Latent Variable Models: Conditional Versus Marginal Likelihoods. *Psychometrika*, 84, 802-829.

[ROB10] Robert, C. P. & Casella, G. 2010. *Introducing Monte Carlo methods with R*, Springer.

[VEH17] Vehtari, A.; Gelman, A. & Gabry, J. 2017. Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and Computing*, 27, 1413-1432.