Estimating abundance in open populations using capture-recapture models

Olivier Gimenez

August 8, 2016

Introduction

Lately, I have found myself repeating the same analyses again and again to estimate population size from capture-recapture models, and the Cormack-Jolly-Seber (CJS) model in particular. I do not intend here to provide extensive details on this model and its variants. It is just a basic attempt to put together some R code to avoid spending hours digging up in my files how to do this analysis.

I use RMark because everything can be done in R, and it's cool for reproducible research. But other pieces of software are fine too. I consider simple CJS models and models with transience. In passing, I also fit models with heterogeneity in the detection process with finite mixtures. The bootstrap is used to obtain confidence intervals.

I have ignored multi-model inference for simplicity. However the bootstrap can be used to perform model selection (e.g., Buckland et al. 1997).

The data and codes are part of a manuscript that is currently in review:

> Chiara G. Bertulli, Loreleï Guéry, Niall McGinty, Ailie Suzuki, Naomi Brannan, Tania Marques, Marianne H. Rasmussen, Olivier Gimenez (in review). Abundance estimation of photographically identified common minke whales, white-beaked dolphins and humpback whales in Icelandic coastal waters using capture-recapture methods.

Abundance estimates from a Cormack-Jolly-Seber model

First, let's load the RMark package for analysing capture-recapture data by calling MARK from R.

```
library(RMark)
```

The data come from an opportunistic monitoring through which photos were taken of Humpback whales in the Icelandic waters. More details in Chiara's paper cited above. Each row is an individual that has been detected (coded as a 1) or non-detected (coded as a 0) through photo-identification over the years in columns. Have a look to the file in your favorite text editor. Other formats are fine.

```
hw.dat = import.chdata("humpbackwhaleMaySep20062013.txt", header = F, field.names = c("ch"), field.types
summary(hw.dat)
```

```
##       ch
##  Length:195
##  Class :character
##  Mode  :character
```

```
attach(hw.dat)
```

Now it is time to build our model. We're gonna use a standard Cormack-Jolly-Seber model. I have carried out goodness-of-fit tests before and found that everything was OK.

```
hw.proc = process.data(hw.dat, model="CJS")
hw.ddl = make.design.data(hw.proc)
```

Then we specify the effects we'd like to consider on survival and detection probabilities.

```
# survival process
Phi.ct = list(formula=~1) # constant
Phi.time = list(formula=~time) # year effect
# detection process
p.ct = list(formula=~1) # constant
p.time = list(formula=~time) # year effect
```

Let's roll and run four models with or without a year effect!

```
# constant survival, constant recapture
Model.1 = mark(hw.proc,hw.ddl,model.parameters=list(Phi=Phi.ct,p=p.ct),output = FALSE,delete=T)
# constant survival, time-dependent recapture
Model.2 = mark(hw.proc,hw.ddl,model.parameters=list(Phi=Phi.ct,p=p.time),output = FALSE,delete=T)
# time-dependent survival, constant recapture
Model.3 = mark(hw.proc,hw.ddl,model.parameters=list(Phi=Phi.time,p=p.ct),output = FALSE,delete=T)
# time-dependent survival, time-dependent recapture
Model.4 = mark(hw.proc,hw.ddl,model.parameters=list(Phi=Phi.time,p=p.time),output = FALSE,delete=T)
```

Now we'd like to check whether there is some heterogeneity in the detection process, because we know that it can lead to bias in abundance estimates. We use a 2-class finite mixture model developped by Shirley Pledger and collaborators. See e.g. Cubaynes et al. (2010).

```
# Model building
hw.proc = process.data(hw.dat, model="CJSMixture")
hw.ddl = make.design.data(hw.proc)
```

We need to add a parameter for the proportion of individuals in each class.

```
# Mixture process
pi.dot=list(formula=~1) # constant
```

Then, we run the same models as above, with heterogeneity.

```
Model.5 = mark(hw.proc,hw.ddl,model.parameters=list(Phi=Phi.ct,p=p.ct,pi=pi.dot),output = FALSE,delete=
Model.6 = mark(hw.proc,hw.ddl,model.parameters=list(Phi=Phi.ct,p=p.time,pi=pi.dot),output = FALSE,delet
Model.7 = mark(hw.proc,hw.ddl,model.parameters=list(Phi=Phi.time,p=p.ct,pi=pi.dot),output = FALSE,delet
Model.8 = mark(hw.proc,hw.ddl,model.parameters=list(Phi=Phi.time,p=p.time,pi=pi.dot),output = FALSE,dele
```

Another way of modelling heterogeneity is to use for individual random effects (Gimenez and Choquet 2010).

Let's have a look to the AIC for these models. Note that it's totally fine to use the AIC to compare models with/without heterogeneity (Cubaynes et al. 2012).

```
# homogeneous models
summary(Model.1)$AICc
```

```
## [1] 317.4295
```

```
summary(Model.2)$AICc
```

```
## [1] 311.3483
```

```
summary(Model.3)$AICc
```

```
## [1] 313.1428
```

```
summary(Model.4)$AICc
```

```
## [1] 322.0564
```

```
# heterogeous models
summary(Model.5)$AICc
```

```
## [1] 319.4569
```

```
summary(Model.6)$AICc
```

```
## [1] 313.2418
```

```
summary(Model.7)$AICc
```

```
## [1] 315.0362
```

```
summary(Model.8)$AICc
```

```
## [1] 323.5249
```

For convenience, we will say that `model 2` is the model best supported by the data, the one with constant survival probability and time-dependent recapture probability. Multi-model selection would be more appropriate here. Let's have a look to the parameter estimates: survival, then recapture probabilities estimates.

```
phitable = get.real(Model.2,"Phi", se= TRUE)
# names(phitable)
phitable[c("estimate","se","lcl","ucl")][1,]
```

```
##                    estimate        se       lcl       ucl
## Phi g1 c1 a0 t1 0.5234884 0.0565077 0.4133879 0.6313524
```

```
ptable = get.real(Model.2,"p", se= TRUE)
ptable[c("estimate","se","lcl","ucl")][1:7,]
```

```
##                  estimate        se       lcl       ucl
## p g1 c1 a1 t2 0.6814077 0.2413566 0.1948450 0.9497567
## p g1 c1 a2 t3 0.1515677 0.1041699 0.0352270 0.4663918
## p g1 c1 a3 t4 0.4246961 0.1541836 0.1764802 0.7177504
## p g1 c1 a4 t5 0.2866319 0.1204953 0.1123644 0.5605063
## p g1 c1 a5 t6 0.3746888 0.1538310 0.1419705 0.6845392
## p g1 c1 a6 t7 0.4332827 0.1215465 0.2246678 0.6685709
## p g1 c1 a7 t8 0.8383462 0.1685125 0.3119202 0.9834243
```

Now it's easy to estimate abundance estimates by calculating the ratios of the number of individuals detected at each occasion over the corresponding estimate of recapture probability. Note that we estimate **re**capture probabilities, so that we cannot estimate abundance on the first occasion.

```
# calculate the nb of recaptured individiduals / occasion
obs = gregexpr("1", hw.dat$ch)
n_obs = summary(as.factor(unlist(obs)))
estim_abundance = n_obs[-1]/ptable$estimate[1:7]
estim_abundance
```

```
##         2         3         4         5         6         7         8
##  33.75365  92.36796  58.86562  45.35434  93.41085 122.32199  90.65467
```

We use a boostrap approach to get an idea of the uncertainty surrounding these estimates, in particular to obtain the confidence intervals.

We first define the number of bootstrap iterations (10 here for the sake of illustration, should be 500 instead, or even 1000 if the computational burden is not too heavy), the number of capture occasions and format the dataset in which we'd like to resample (with replacement). This is non-parametric bootstrap (and alternative is parametric bootstrap where data are simulated using the model estimates). We also define a matrix `popsize` in which we will store the results, and we define the seed for simulations (to be able to replicate the results).

```
nb_bootstrap = 10
nb_years = 8
target = data.frame(hw.dat,stringsAsFactors=F)
popsize = matrix(NA,nb_bootstrap, nb_years-1)
set.seed(5)
pseudo = target # initialization
```

Finally, we define the model structure and the effects on parameter (same for all bootstrap samples).

```
# define model structure
hw.proc = process.data(pseudo, model="CJS")
hw.ddl = make.design.data(hw.proc)
# define parameter structure
phi.ct = list(formula=~1)
p.time = list(formula=~time)
```

Let's run the bootstrap now:

```
for (k in 1:nb_bootstrap){
  # resample in the original dataset with replacement
  pseudo$ch = sample(target$ch, replace=T)
  # fit model with Mark
  res = mark(hw.proc,hw.ddl,model.parameters=list(Phi=phi.ct,p=p.time),delete=TRUE,output=FALSE)
  # get recapture prob estimates
  ptable = get.real(res,"p", se= TRUE)
  # calculate the nb of recaptured individiduals / occasion
  allobs = gregexpr("1", pseudo$ch)
  n = summary(as.factor(unlist(allobs)))
  popsize[k,] <- n[-1]/ptable$estimate[1:(nb_years-1)]
}
```

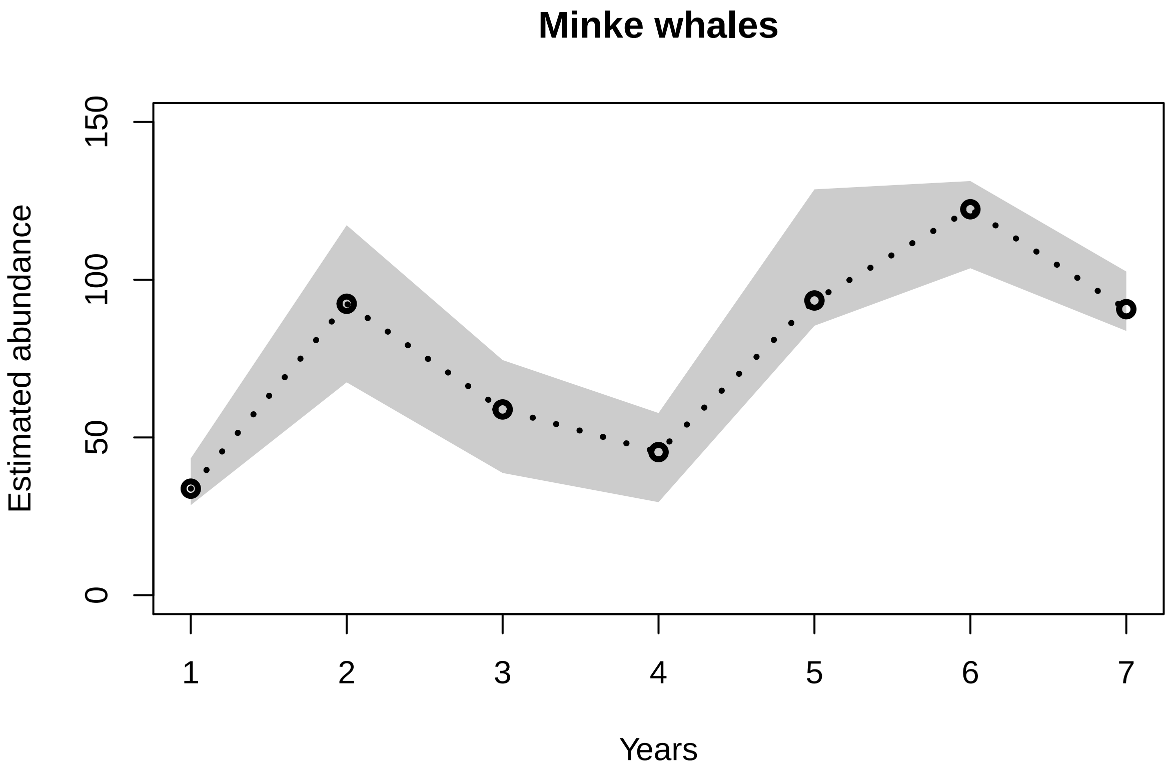
Now we can get confidence intervals:

```
ci_hw = apply(popsize,2,quantile,probs=c(2.5/100,97.5/100),na.rm=T)
ci_hw
```

```
##             [,1]      [,2]     [,3]     [,4]      [,5]     [,6]      [,7]
## 2.5%   28.54388  67.46157 38.73358 29.48032  85.40419 103.6275  83.76611
## 97.5% 43.36611 117.27431 74.52388 57.73956 128.64005 131.2654 102.58292
```

A plot:

```
plot(1:(nb_years-1),estim_abundance, col="black", type="n", pch=21, xlab="Years", lty=3, ylab="Estimated
polygon(c(rev(1:(nb_years-1)), 1:(nb_years-1)), c(rev(ci_hw[2,]), ci_hw[1,]), col = 'grey80', border = 
lines(1:(nb_years-1), estim_abundance, col="black",lty=3,type='o',lwd=3,pch=21)
```



What if transience occurs?

We now analyse data on Minke whales.

```
library(RMark)
mw.dat = import.chdata("minkewhalesAprAug20082013.txt", header = F, field.names = c("ch"), field.types
summary(mw.dat)
```

```
##       ch
##  Length:191
##  Class :character
##  Mode  :character
```

```
attach(mw.dat)
```

The goodness-of-fit tests showed that Test3SR was significant, hence a transient effect due to true transient individuals, an age effet or a bit of both. To account for this effect, we use a two-age class structure on survival. We also consider a model structure with heterogeneity as in the previous section.

```
# homogeneous structure
mw.proc = process.data(mw.dat, model = "CJS")
mw.ddl = make.design.data(mw.proc)
mw.ddl = add.design.data(mw.proc, mw.ddl,"Phi", type = "age", bins = c(0,1,6), name = "ageclass", right
# heterogeneous structure
mw.proc2 = process.data(mw.dat, model="CJSMixture")
mw.ddl2 = make.design.data(mw.proc2)
mw.ddl2 = add.design.data(mw.proc2, mw.ddl2,"Phi", type = "age", bins = c(0,1,6), name = "ageclass", ri
```

Then we specify the effects on survival and detection probabilities. Age is always included. We consider time-dependent variation or not on both survival and recapture probabilities.

```
# survival process
phi.age = list(formula=~ageclass)
phi.ageptime = list(formula=~ageclass+time)
# detection process
p.ct = list(formula=~1)
p.time = list(formula=~time)
# mixture process
pi.dot = list(formula=~1)
```

Let's roll and run models with or without a year effect, and with or without heterogeneity in the recapture probability!

```
Model.1 = mark(mw.proc, mw.ddl, model.parameters = list(Phi = phi.age, p = p.ct),output = FALSE,delete=
Model.2 = mark(mw.proc, mw.ddl,model.parameters = list(Phi = phi.age, p = p.time),output = FALSE,delete
Model.3 = mark(mw.proc, mw.ddl,model.parameters = list(Phi = phi.ageptime, p = p.ct),output = FALSE,del
Model.4 = mark(mw.proc, mw.ddl,model.parameters = list(Phi = phi.ageptime, p = p.time),output = FALSE,d
Model.5 = mark(mw.proc2,mw.ddl2,model.parameters = list(Phi = phi.age, p = p.ct, pi = pi.dot),output =
Model.6 = mark(mw.proc2,mw.ddl2,model.parameters = list(Phi = phi.age, p=p.time, pi = pi.dot),output =
Model.7 = mark(mw.proc2,mw.ddl2,model.parameters = list(Phi = phi.ageptime, p = p.ct, pi = pi.dot),outp
Model.8 = mark(mw.proc2,mw.ddl2,model.parameters = list(Phi = phi.ageptime, p = p.time, pi = pi.dot),ou
```

Let's have a look to the AIC for these models.

```
# homogeneous models
summary(Model.1)$AICc
```

```
## [1] 480.6304
```

```
summary(Model.2)$AICc
```

```
## [1] 486.8038
```

```
summary(Model.3)$AICc
```

```
## [1] 485.1968
```

```
summary(Model.4)$AICc
```

```
## [1] 493.2464
```

```
# heterogeneous models
summary(Model.5)$AICc
```

```
## [1] 482.672
```

```
summary(Model.6)$AICc
```

```
## [1] 488.8501
```

```
summary(Model.7)$AICc
```

```
## [1] 487.2431
```

```
summary(Model.8)$AICc
```

```
## [1] 495.2567
```

For simplicity here, we will say that `model 1` is the model that is best supported by the data, the one with constant survival and recapture probabilities. Let's have a look to the parameter estimates: survival, then recapture probabilities estimates.

```
phitable = get.real(Model.1,"Phi", se= TRUE)
# names(phitable)
phitable[c("estimate","se","lcl","ucl")][1:2,]
```

```
##                    estimate        se       lcl       ucl
## Phi g1 c1 a0 t1 0.4313735 0.0507980 0.3357811 0.5323679
## Phi g1 c1 a1 t2 0.7976824 0.0513587 0.6787706 0.8803360
```

On the first row, the survival is for both transient and resident individuals. On the second row, this is survival for resident individuals.

```
ptable = get.real(Model.1,"p", se= TRUE)
ptable[c("estimate","se","lcl","ucl")][1,]
```

```
##                 estimate        se       lcl      ucl
## p g1 c1 a1 t2 0.5353636 0.0536269 0.4302436 0.637433
```

An estimate of abundance is obtained as in the previous section:

```
# calculate the nb of recaptured individiduals / occasion
obs = gregexpr("1", mw.dat$ch)
n_obs = summary(as.factor(unlist(obs)))
estim_abundance = n_obs[-1]/ptable$estimate[1]
estim_abundance
```

```
##         2         3         4         5         6
##  72.84769 115.80914  98.99814  76.58347  78.45136
```

We use a boostrap approach to get an idea of the uncertainty surrounding these estimates, in particular to obtain the confidence intervals. The bootstrap approach was proposed by Madon et al. (2013). Roger Pradel discovered a bug in the appendix that he corrected. He also substantially simplified the code. I found a minor problem in Roger's code that I corrected. I know, version control would be great…

We first define a few quantities. See previous section for details.

```
nb_bootstrap = 10
nb_years = 6

target = data.frame(mw.dat,stringsAsFactors=F)
pseudo = target # initialization

popTot = popT = popR = matrix(NA, nb_bootstrap, nb_years-1) # abundance
tau = rep(NA, nb_bootstrap) # transient rate
det.p = rep(NA, nb_bootstrap) # recapture

set.seed(5)

# model structure
mw.proc <- process.data(mw.dat, model = "CJS")
mw.ddl <- make.design.data(mw.proc)
mw.ddl <- add.design.data(mw.proc, mw.ddl,"Phi", type = "age", bins = c(0,1,6), name = "ageclass", righ

# parameters
phiage <- list(formula=~ageclass)
p.ct <- list(formula=~1)
```

Let's run the bootstrap now:

```
for (k in 1:nb_bootstrap){
  # draw new sample
  pseudo$ch = sample(target$ch, replace=T)
  # calculate R and m
  firstobs = regexpr("1", pseudo$ch)
  R = summary(factor(firstobs,levels=1:nb_years))
  allobs = gregexpr("1", pseudo$ch)
  n = summary(as.factor(unlist(allobs)))
  m = n-R
  # fit model with 2 age classes on survival and constant recapture with MARK
  phiage.pct = mark(process.data(pseudo),mw.ddl,model.parameters=list(Phi=phiage,p=p.ct),output = FALSE
  tau[k] = 1 - phiage.pct$results$real[1,1] / phiage.pct$results$real[2,1] # transient rate
  det.p[k] = phiage.pct$results$real[3,1]
  # calculate abundance of residents and transients
  popR[k,] = (m[-1] + R[-1] * (1 - tau[k])) / det.p[k]
  popT[k,] = R[-1] * tau[k] / det.p[k]
}
```

Now we can calculate the abundance of residents and a confidence interval:

```
popRmean = apply(popR,2,mean) # mean resident population size
popRmean
```

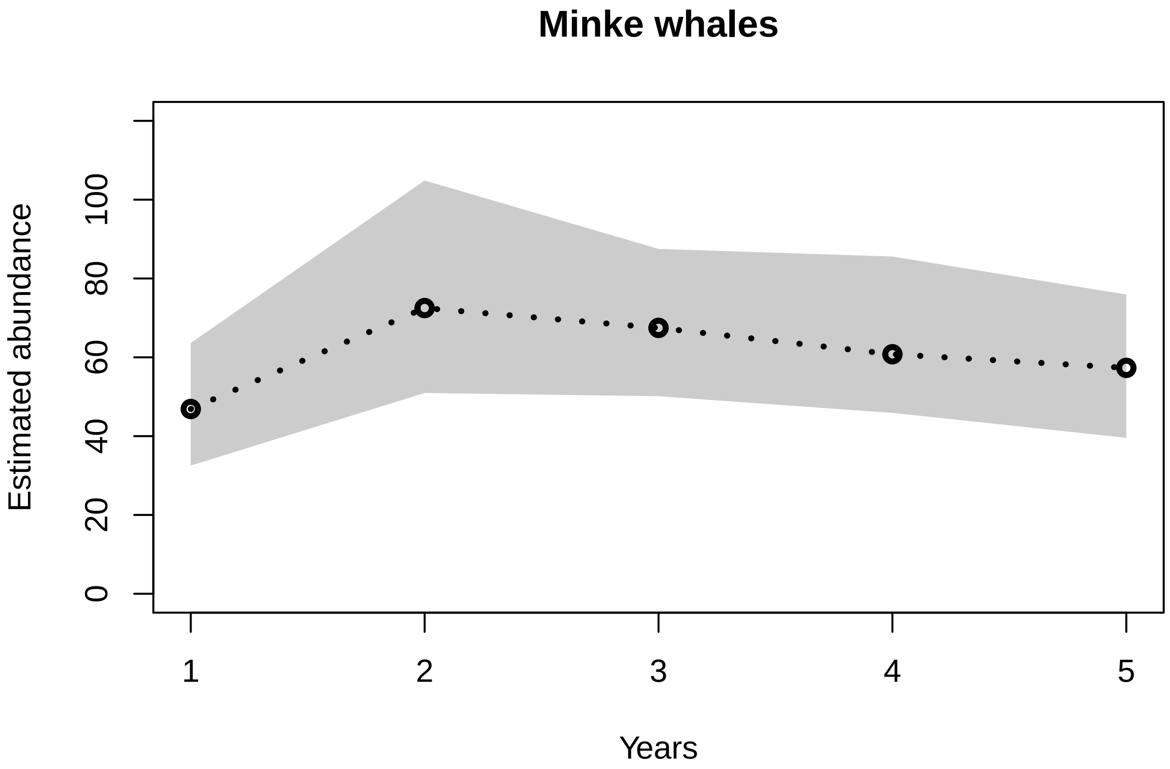
```
## [1] 46.88135 72.49815 67.46543 60.77500 57.31774
```

```
popRci = apply(popR,2,quantile,c(0.025,0.975))
popRci
```

```
##            [,1]      [,2]     [,3]     [,4]     [,5]
## 2.5%  32.52640  50.95279 50.12659 45.89512 39.58013
## 97.5% 63.62993 104.86693 87.50820 85.56882 75.93378
```

A plot:

```
plot(1:(nb_years-1),popRmean, col="black", type="n", pch=21, xlab="Years", lty=3, ylab="Estimated abunda
polygon(c(rev(1:(nb_years-1)), 1:(nb_years-1)), c(rev(popRci[2,]), popRci[1,]), col = 'grey80', border
lines(1:(nb_years-1), popRmean, col="black",lty=3,type='o',lwd=3,pch=21)
```



Lastly, it is also possible to calculate an estimate of the transient rate along with its confidence interval using the bootstrap. Alternatively, the delta-method could be used.

```
mean(tau)
```

```
## [1] 0.4827876
```

```
quantile(tau,probs=c(2.5,97.5)/100)
```

```
##      2.5%     97.5% 
## 0.3758783 0.6283435
```

To do

- multi-model inference using bootstrap à la Buckland
- add complete reference for Chiara's paper once published
- models with individual random effects
- wolf example to illustrate abundance estimation with heterogeneity
- add Jolly-Seber as in Karamanlidis et al. (2015) ; add robust-design as in Nina and Blaise papers.

References

- Buckland ST, Burnham KP, Augustin NH (1997). Model selection: An integral part of inference. Biometrics. 53:603–618.

- Cubaynes, S., C. Lavergne, E. Marboutin, and O. Gimenez (2012). Assessing individual heterogeneity using model selection criteria: How many mixture components in capture-recapture models? Methods in Ecology and Evolution 3: 564-573.

- Cubaynes, S. Pradel, R. Choquet, R. Duchamp, C. Gaillard, J.-M., Lebreton, J.-D., Marboutin, E., Miquel, C., Reboulet, A.-M., Poillot, C., Taberlet, P. and O. Gimenez. (2010). Importance of accounting for detection heterogeneity when estimating abundance: the case of French wolves. Conservation Biology 24:621-626.

- Gimenez, O. and R. Choquet (2010). Incorporating individual heterogeneity in studies on marked animals using numerical integration: capture-recapture mixed models. Ecology 91: 951-957.

- Karamanlidis, A.A., M. de Gabriel Hernando, L. Krambokoukis, O. Gimenez (2015). Evidence of a large carnivore population recovery: counting bears in Greece. Journal for Nature Conservation. 27: 10–17

- Madon, B., C. Garrigue, R. Pradel, O. Gimenez (2013). Transience in the humpback whale population of New Caledonia and implications for abundance estimation. Marine Mammal Science 29: 669-678.