Introduction to R for distance sampling

Workshop, 26 August 2018

Centre for Research into Ecological and Environmental Modelling

1.4 Fitting GLMs, prediction and bootstrapping

This practical is based on the lure point transect case study in Buckland *et al.* (2015), Section 10.2.1, which is a simplified version of the analysis in Summers and Buckland (2011). Generalised linear models (GLMs) are used to model the response of Scottish crossbills to a lure in order to estimate their probability of response and hence estimate the density and abundance. To provide a measure of precision for the abundance estimate, 95% confidence intervals are obtained by bootstrapping.

The Scottish crossbill (*Loxia scotica*) is Britain's only endemic bird species. A point transect study was conducted to obtain the number of birds within each point after responding to an audible lure. The probability of responding to the lure was estimated by recording the response of previously detected birds to the lure at different distances (Summers and Buckland 2011).

Objectives of the practical

1. Fit a GLM
2. Obtain predicted values from GLM
3. Calculate abundance
4. Using `for` loops to bootstrap abundance.

The lure trials

The data provided in the response trials are:

- No. - trial number
- day - days from 1st January
- time - hour of the day
- habitat - habitat type (1=plantation, 2= native pinewood)
- dist - distance of the bird when the lure was played (m)

- behavcode - behaviour code (1=perching and feeding, 2= giving excitement calls, 3=singing)
- numbirds - flock size
- response - response of bird to lure (0=no response, 1=response).

The trials data are in file `lure trials.csv`. We first import the data and check that it has been read in correctly:

```
xbill <- read.csv("lure-trials.csv",header=T)
```

```
head(xbill)
```

Summarising the data

We can see how many birds did, or did not, respond to the lure:

```
table(xbill$response)
```

To add totals to the table:

```
addmargins(table(xbill$response))
```

There are six potential covariates that might affect the probability that a bird responds to the lure (`day`, `time`, `dist`, `numbirds`, `habitat`, and `behavcode`): the latter two are factor type variables and so we need to indicate to R to treat them as factors:

```
# Specify variables as factors
xbill$habitat <- factor(xbill$habitat)
xbill$behavcode <- factor(xbill$behavcode)
```

To look at the response in each factor level we can obtain a two-way table, for example:

```
addmargins(table(xbill$response, xbill$habitat))
```

A boxplot could be used to look at the distribution of distances for each response level - refer to Practical 1.3 if you are not sure how to do this.

Fitting a GLM

We want to explain the probability of response in terms of the potential covariates. The dependent variable, `response`, can only take two values (0 and 1) and so rather than fit a linear regression model to these data we fit a GLM. The `glm` function allows us to specify a distribution for the dependent variable in the model with the `family` argument.

We can include all the covariates in a model as follows.

```
# Fit a model with all potential covars
model1 <- glm(response~dist+numbirds+day+time+habitat+behavcode, family=binomial,
              data=xbill)
```

As usual, the `summary` function can be used to display details of the model object.

```
summary(model1)
```

We see that only `dist` has a coefficient that is significantly different from zero. Experiment with dropping non-significant terms. Using a backwards stepping procedure, you should find that other covariates remain non-significant, so we can use the simple model:

```
# Fit model with dist only
model2 <- glm(response~dist, family=binomial, data=xbill)
```

```
summary(model2)
```

(Degrees of freedom change a little between models because some covariates have a few missing values and so these observations are excluded.)

Prediction

Having fitted a model, we now want to see how the predicted probability of response changes with distance (similar to a detection function model). We assume that the maximum distance that a crossbill will respond to a lure is 850m. Here we create a 'prediction' data frame that has one column called `dist` and this ranges from 0 to 850 (in unit intervals). (Prediction data needs to contain objects with the same names as the explanatory variables in the fitted model.)

```
# Specify maximum distance
w <- 850
# Create data frame containing distances
preddata <- data.frame(dist=0:w)
# Predicton
phat <- predict.glm(model2, newdata=preddata, type="response")
```

Now we have the estimated probabilities, we can overlay this as a red line onto a plot of the observed responses (black circles). The `xlim` argument specifies the limits of the x-axis:

```
plot(xbill$dist, xbill$response, xlim=c(0, w), xlab="Distance from lure (m)",
     ylab="Probability of response")
lines(preddata$dist, phat, col="red")
```

Estimating abundance

This section gets a bit technical and so don't worry too much about the details if you are not familiar with point transects.

Abundance is obtained from

$$\hat{N} = \frac{n.A}{P_a.a}$$

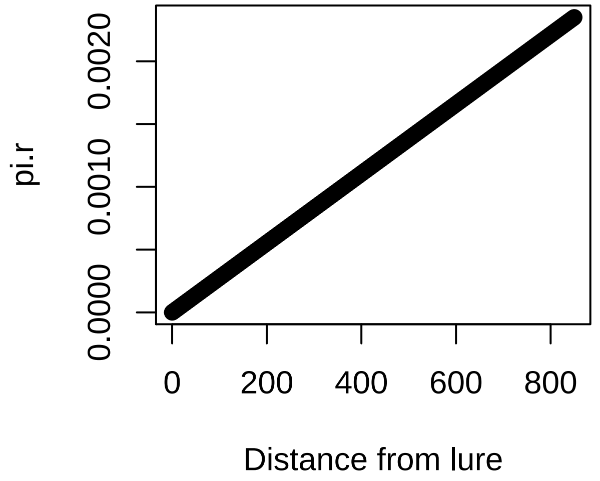
where

- n is the number of detections
- A is the area of the study region (i.e. 3505.8 km^2)
- P_a is the probability of response (or detection) in the covered area, and
- a is the area of the covered region (i.e. $a = k\pi w^2$ where k is the number of points)

First we calculate P_a. To do this we need to specify the function $\pi(r), r \leq w$, which represents the probability density function (pdf) of distances of animals from the point. If we make some assumptions, we can think of the pdf as triangular:

```r
# Create pdf
pi.r <- preddata$dist/sum(preddata$dist)

# Plot pdf
plot(preddata$dist, pi.r, xlab="Distance from lure")
```



Then we multiply $\pi(r)$ with the probability of response and (numerically) integrate from 0 to w.

```
# Calculate Pa
Pa <- sum(phat * pi.r)
Pa
```

```
## [1] 0.09709381
```

Assuming that the probability of response is a function of distance only (as in `model2`), then P_a is just less than 10% (i.e. <10% of birds within 850m of a point are detected).

Now we need to calculate the other parameters; for this we need to read in data from the point transect survey. These data consist of:

- point - point transect identifier
- nscottish - the number of Scottish crossbills detected at the point

Note that detection distances are unknown in the main survey: instead, we have used the trials data to estimate the detection function, and hence the proportion of birds within 850m that are detected.

```
# Read in detections
detections <- read.csv("mainsurveydetections.csv", header=T)

# Total number of Scottish crossbills detected
n <- sum(detections$nscottish)
```

We now calculate the number of points (k) in the main survey, and hence, the total covered area within 850m of a point, converting from m^2 to km^2. Note that `pi` is a reserved word to represent π (i.e. 3.141593).

```
# Number of points
k <- length(detections$point)

# Covered area (km2)
a <- k * pi * (w/1000)^2

# Size of the study region (km2)
A <- 3505.8
```

We can now estimate the size of the population as:

```
Nscot <- (n*A)/(Pa*a)
```

What is your estimate of the population of Scottish crossbills?

Confidence intervals

We can calculate 95% confidence interval for true abundance by bootstrapping both trials and points. The steps involved are

1. randomly generate (with replacement) a new set of response data
2. estimate P_a for the new data
3. generate (with replacement) a new set of point transect data
4. estimate abundance
5. repeat steps 1-4 many times to build a distribution of abundances
6. take the 2.5 and 97.5 percentiles of the distribution.

The following code does this:

```r
# Initialise parameters
# Number of bootstraps
nboot <- 1000
# Number of trials
m <- length(xbill$dist)
# Create empty vectors to store new sample
bdist <- vector(length=m)
bresponse <- vector(length=m)
# Create empty vector to store bootstrap abundances
bNscot <- vector(length=nboot)
# Create prediction data
pred <- data.frame(bdist=0:w)

# Start bootstrap loop
for (i in 1:nboot) {
  # Bootstrap trials
  # Generate index of sample
  btindex <- sample(1:m, size=m, replace=TRUE)
  for (j in 1:m) {
    bdist[j] <- xbill$dist[btindex[j]]
    bresponse[j] <- xbill$response[btindex[j]]
  }
  # Fit GLM
  bmodel <- glm(bresponse ~ bdist, family=binomial)
  # Predict probability of response
  bphat <- predict.glm(bmodel, newdata=pred, type="response")
  # Calculate Pa
  bPa <- sum(bphat * pi.r)
  # Bootstrap points
  rindex <- sample(1:k, k, replace=TRUE)
  n <- sum(detections$nscottish[rindex])

  # Calculate abundance
  bNscot[i] <- (n*A)/(bPa*a)
} # End of bootstrap loop
```

Having obtained a distribution of abundances, the 2.5 and 97.5 percentiles can be obtained easily:

```r
# Obtain percentiles
quantile(bNscot, probs=c(0.025, 0.975))
```

References

Buckland ST, Rexstad EA, Marques TA and Oedekoven CS (2015) Distance Sampling: Methods and Applications. Springer 277 pp. ISBN: 978-3-319-19218-5 (Print) 978-3-319-19219-2 (Online)

Buckland ST, Summers RW, Borchers DL and Thomas L (2006) Point transect sampling with traps or lures. Journal of Applied Ecology 43: 377-384

R Core Team (2017) R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/

Summer RW and Buckland ST (2011) A first survey of the global population size and distribution of the Scottish Crossbill *loxia scotia*. Bird Conservation International 21: 186-198