Distance sampling online workshop

Analysis in R: Covariates in the detection function

CREEM, Univ of St Andrews – October 2018

1 Using survey data

In this practical, we use some real data to fit different detection function models and estimate density and abundance. The data were collected during line transect surveys of duck nests in Monte Vista National Wildlife Refuge, Colorado, USA in 1967 and 1968. Twenty transects, each 25.75km in length were walked 5 times over the two years. Total transect length of 128.75km (25.75 $\times$ 5) and a distance out to 2.4m was searched. Consult Anderson and Pospahala (1970) for a description of the survey. Distances of detected nests have been provided in a 'csv' text file in a basic format required by 'Distance'. The columns in the the file are:

- Study.Area - name of the study region (Monte Vista NWR)
- Region.Label - identifier of regions or strata (in this case there is only one region and it is set to 'Default')
- Area - size of the stratum
- Sample.Label - line transect identifier
- Effort - length of each transect
- distance - perpendicular distances (m).

The distances allow different key functions/adjustments to be fitted in the detection function model and, by including the transect lengths and area of the region, density and abundance can be estimated.

1.1 Objectives of the practical

1. Import a text file
2. Understand the structure of a data frame
3. Fit different key functions/adjustments in the detection function model
4. Explore the model object i.e. `ddf` and `dht`
5. Create a data frame
6. Estimate density and abundance using `ds`.

1.2 Importing the data

To let R (R Core Team, 2018) know where to look for files (and also where to save the R workspace and '.Rmd' files) we can set the 'working directory'; from the menu along the top of the `RStudio` window click on 'Session > Set Working Directory > Choose Directory' and select your chosen directory, for example 'C:/workshop'.

Load the data into R with the following command:

```
nests <- read.csv(file = "datasets/ducks-area-effort.csv",
    header = TRUE)
```

This command is made up of several components:

- `read.csv` is a function to read data files of type 'csv' (comma-separated values),
- the function has two arguments specified; `file` specifies the name of the data file and `header=TRUE` specifies that the first row of the data file contains the names of the data columns.
- the `<-` symbol has assigned the data set to an object called `nests`. Note that there is now an object called `nests` listed on the 'Environment' tab.

To check that the data file has been read into R correctly, use the `head` and `tail` 'functions' to look at the top and bottom rows of the data, respectively. To look at the first few rows of `nests` type the following command.

```
head(nests)
```

```
##   Region.Label  Area Sample.Label Effort
## 1      Default 40.47            1 128.75
## 2      Default 40.47            1 128.75
## 3      Default 40.47            1 128.75
## 4      Default 40.47            1 128.75
## 5      Default 40.47            1 128.75
## 6      Default 40.47            1 128.75
##   distance
## 1     0.06
## 2     0.07
## 3     0.04
## 4     0.01
## 5     0.37
## 6     0.36
```

The `head` function as used above displays the first 6 records of the named object. By default, `head` and `tail` display 6 rows of data but this can be changed by specifying a value for the function argument which controls this action. To display the last 2 records in the data, type the command:

```
tail(nests, n = 2)
```

```
##     Region.Label  Area Sample.Label Effort
## 533      Default 40.47           20 128.75
## 534      Default 40.47           20 128.75
```

```
##     distance
## 533     2.38
## 534     2.13
```

In this function, `n` is the argument which controls the number of rows to display.

The object `nests` is a dataframe object made up of rows and columns. Use the function `dim` to find out the dimensions of the data set (i.e. the total number of rows and columns):

```
dim(nests)
```

```
## Number of rows 534  Number of columns  5
```

Another way to look at a data frame is to move to the 'Environment tab' and click on the rectangle (with the grid); this opens a new tab showing the data.

1.3 Summarising the perpendicular distances

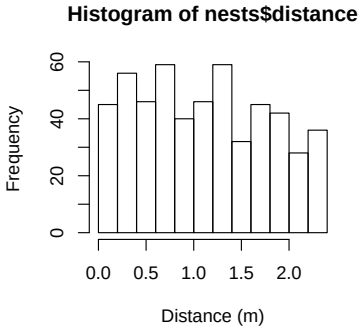
To access an individual column within a data frame we use the `$` symbol, for example to summarise the distances, then the following command is used:

```
summary(nests$distance)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.
##   0.010   0.540   1.080   1.117   1.670
##    Max.
##   2.400
```

Similarly to plot the histogram of distances, the command is:

```
hist(nests$distance, xlab = "Distance (m)")
```



1.4 Fitting different models

To use the `ds` function, first ensure that the `Distance` package (Miller, 2017) has been loaded.

The function `ds` requires a data frame to have a column called `distance`, we specify the name of the data frame as follows:

```
nest.model1 <- ds(nests, key = "hn", adjustment = NULL)
```

```
## Fitting half-normal key function
```

```
## Key only model: not constraining for monotonicity.
```

```
## AIC= 928.134
```

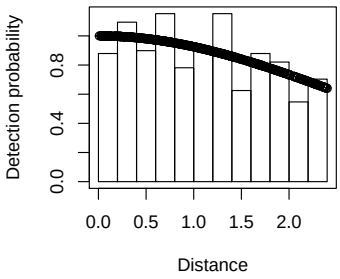
In this command, a half-normal key function is selected with no adjustment terms. Summarise the fitted model:

```
summary(nest.model1$ddf)
```

```
##
## Summary for ds object
## Number of observations :  534
## Distance range         :  0  -  2.4
## AIC                    :  928.1338
##
## Detection function:
##  Half-normal key function
##
## Detection function parameters
## Scale coefficient(s):
##               estimate        se
## (Intercept) 0.9328967 0.1703933
##
##                         Estimate          SE
## Average p              0.8693482  0.03902053
## N in covered region 614.2533225 29.19683067
##                               CV
## Average p            0.04488481
## N in covered region  0.04753223
```

Plot the detection function with the histogram having 12 bins:

```
plot(nest.model1, nc = 12)
```



To fit different detection function shapes, we can change the `key` and `adjustment` arguments. For example to fit a half-normal key function with cosine adjustment terms, then use the command:

```
nest.model2 <- ds(nests, key = "hn", adjustment = "cos")
```

By default, AIC selection will be used to fit adjustment terms of up to order 5. Have any adjustment terms been selected?

```
##
## Summary for ds object
## Number of observations :  534
## Distance range         :  0  -  2.4
## AIC                    :  928.1338
```

```
##
## Detection function:
##  Half-normal key function
##
## Detection function parameters
## Scale coefficient(s):
##              estimate        se
## (Intercept) 0.9328967 0.1703933
##
##                         Estimate          SE
## Average p              0.8693482  0.03902053
## N in covered region 614.2533225 29.19683067
##                              CV
## Average p           0.04488481
## N in covered region 0.04753223
```

To fit a hazard rate key function with Hermite polynomial adjustment terms, then use the command:

```
nest.model3 <- ds(nests, key = "hr", adjustment = "herm")
```

```
summary(nest.model3$ddf)
```

```
##
## Summary for ds object
## Number of observations :  534
## Distance range         :  0  -  2.4
## AIC                    :  929.7934
##
## Detection function:
##  Hazard-rate key function
##
## Detection function parameters
## Scale coefficient(s):
##              estimate        se
## (Intercept) 0.9190194 0.2081042
##
## Shape coefficient(s):
##              estimate        se
## (Intercept) 0.2899026 0.6393387
##
##                         Estimate          SE
## Average p              0.8890698  0.04958136
## N in covered region 600.6277685 34.59620441
##                              CV
```

```
## Average p           0.05576768
## N in covered region 0.05760007
```

Use the `help` command to find out what other key functions and adjustment terms are available.

No, there is barely any difference between the estimated probability of detection. For the first two models, P_a_= `r round(nest.model1$ddf$fitted[1],3)` and for the hazard rate model, P_a_ = `r round(nest.model3$ddf$fitted[1],3)`.

1.5 The `ds` object

The objects created with `ds` (e.g. `nest.model1`) are made up of several parts. We can list them using the `names` function as below:

```
names(nest.model1)
```

```
## [1] "ddf" "dht"
```

The detection function information is in the `ddf` part and the density and abundance estimates would be stored in the `dht` part. To access each part, then the `$` can be used (as with columns in a data frame). For example to see what information is stored in the `ddf` part, we can use the `names` function again:

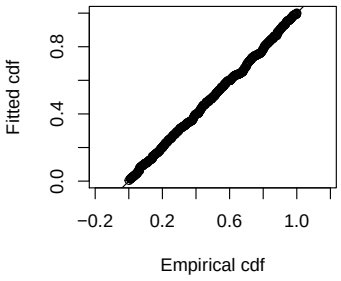
```
names(nest.model1$ddf)
```

```
##  [1] "call"         "data"
##  [3] "model"        "meta.data"
##  [5] "control"      "method"
##  [7] "ds"           "par"
##  [9] "lnl"          "hessian"
## [11] "dsmodel"      "criterion"
## [13] "fitted"       "Nhat"
## [15] "name.message"
```

1.6 Goodness of fit

The usual tools for checking goodness of fit are available, for example:

```
gof_ds(nest.model1)
```




```
##
## Goodness of fit results for ddf object
##
## Distance sampling Cramer-von Mises test (unweighted)
## Test statistic = 0.0353634 p-value = 0.955416
```

Interpretation of qq plot as well as CvM test results

1.7 Estimating density and abundance

this is overly complicated, simply provide ducknests.csv with area and transect lengths from the outset–practicioners will have this information in their own data

No need to refit the model, the results already exist, simply need to be described

So far, we have concentrated on the detection function but, with more information such as transect lengths and the area of the region, we can estimate density and abundance.

```
nest.model1$dht
```

```
##
## Summary statistics:
##    Region  Area CoveredArea Effort   n  k
## 1 Default 40.47       12360   2575 534 20
##          ER       se.ER      cv.ER
## 1 0.2073786 0.007970756 0.03843576
##
## Abundance:
##   Label Estimate        se         cv
## 1 Total 2.011232 0.1188493 0.05909276
##        lcl      ucl       df
## 1 1.788907 2.261188 99.55689
##
## Density:
##   Label   Estimate          se         cv
## 1 Total 0.04969687 0.002936725 0.05909276
##         lcl        ucl       df
## 1 0.0442033 0.05587318 99.55689
```

Now we have all the necessary information to estimate density and abundance, therefore, we can include these tables in the `ds` function:

```
nest.model4 <- ds(nests, key = "hn", adjustment = NULL,
    convert.units = 0.001)
```

The `convert.units` argument ensures that the correct units are specified - in this example, distances are in metres, lengths in km and the area in km^2.

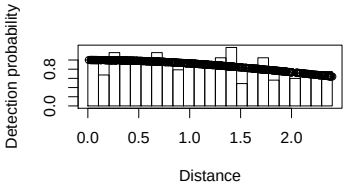
Having run the function, the estimates are stored in the `dht` part of the object:

```
# Print out estimates
nest.model4$dht
```

```
##
## Summary statistics:
##    Region  Area CoveredArea Effort   n  k
## 1 Default 40.47       12.36   2575 534 20
##          ER       se.ER      cv.ER
## 1 0.2073786 0.007970756 0.03843576
##
## Abundance:
##   Label Estimate       se         cv
## 1 Total 2011.232 118.8493 0.05909276
##        lcl      ucl       df
## 1 1788.907 2261.188 99.55689
##
## Density:
##   Label Estimate       se         cv     lcl
## 1 Total 49.69687 2.936725 0.05909276 44.2033
##        ucl       df
## 1 55.87318 99.55689
```

describe each component table of output

```
plot(nest.model4)
```



References

Anderson, D. R. and R. S. Pospahala. 1970. Correction of bias in belt transect studies of immotile objects. The Journal of Wildlife Management, **34**:141–146. URL http://www.jstor.org/stable/3799501.

Buckland, S. T., E. A. Rexstad, T. A. Marques, and C. S. Oedekoven. 2015. Distance Sampling: Methods and Applications. Springer. URL https://www.springer.com/gb/book/9783319192185.

Miller, D. L. 2017. Distance: Distance Sampling Detection Function and Abundance Estimation. URL https://CRAN.R-project.org/package=Distance. R package version 0.9.7.

R Core Team. 2018. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.