Distance sampling online workshop

Analysis in R: Analysis of multi-species surveys

CREEM, Univ of St Andrews – October 2018

European robin *(Erithacus rubecula)*; one of the species in the Montrave study of Buckland (2006).

1 More complex analyses

This practical is based on the Montrave songbird case study in Buckland et al. (2015, Ch. 5), with computer code under Montrave songbird case study. Both point and line transect surveys were conducted and here we use the data from the **line transect** data, although the issues (and solutions) will be similar.

These data are provided in a 'flat file' format (i.e. it contains all the necessary columns to estimate a detection function, density and abundance). While both formats are equally valid, the 'flat file' approach has a particular idiosyncrasy which we exploit here to introduce more functions and data manipulation.

Several species of birds were identified but not all species were detected on all transects. If a simple data selection is performed to select records for a particular species, then not all of the transects will be included in the resulting data (because that species may not have been seen). This doesn't matter if we are only interested in fitting detection functions, but will matter if we wish to estimate density and abundance because the effort will be too low since some of the transects are missing. To correct for this, some data frame manipulation is required. There is generally more than one way to do something in R (R Core Team, 2018) - for an alternative way see the computer code 'Montrave song bird case study' associated with Buckland et al. (2015).

1.1 Objectives of the practical

1. Data frame selection and manipulation
2. Extracting estimates from `dht` object
3. Customising detection function plots

1.2 Importing the data

The data is in a 'flat file' format and contains the following columns:

- Region.Label - name of study
- Area - size of study region (km^2)
- repeats - number of visits to transect
- Sample.Label - line transect identifier
- Effort - length of transect (km)
- distance - perpendicular distance (m)
- species - species of bird (c=chaffinch, g=great tit, r=robin and w =wren)
- visit - on which visit bird was detected.

Use the following command to import the data and then use the `head` command to ensure it has been imported correctly.

```
birds <- read.csv("datasets/montrave-line.csv",
    header = TRUE)
```

```
head(birds, n = 2)
```

```
##   Region.Label Area repeats Sample.Label
## 1     Montrave 33.2       2            1
## 2     Montrave 33.2       2            1
##   Effort distance species visit
## 1  0.208       75       c     1
## 2  0.208       40       c     1
```

Question: Explore the data. How many transects are there?

```
length(unique(birds$Sample.Label))
```

```
## [1] 19
```

For now, save the transect labels to a new object as we will use them later on:

```
tran.lab <- unique(birds$Sample.Label)
```

The `table` command is a quick way to determine how many detections there are of each species:

```
table(birds$species)
```

```
##
##   c   g   r   w
##  73  32  82 156
```

As a hint of things to come, create a two-way table showing the number of detections by transect and by species. If there are zeroes in this table, it will create a challenge.

```
with(birds, table(species, Sample.Label))
```

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
c	4	7	7	5	9	7	5	1	1	3	1	0	2	2	4	3	4	7	1
g	0	2	3	5	5	1	3	2	1	1	0	0	2	0	2	0	3	1	1
r	3	8	11	5	8	7	5	7	3	1	0	2	5	6	4	0	4	3	0
w	10	11	12	11	14	12	13	11	6	3	1	4	9	12	9	2	7	6	3

Each of the line transects was visited twice which is not taken into account at present. However, it is straightforward to do so:

```
birds$Effort <- birds$Effort * birds$repeats
```

1.3 Manipulating the robin data

For the purposes of this practical, we are interested in estimating the density of robins and so we select only these records:

```
robins <- birds[birds$species == "r", ]
```

Question: On how many transects were robins detected?

```
length(unique(robins$Sample.Label))
```

```
## [1] 16
```

If we were to use the `robins` data as it is at present to estimate density, then density would be **incorrect** because the search effort associated with three transects is missing. Adding these missing transects to the `robins` data, requires several steps:

1. identify the missing transects,
2. select the information for the missing transects,
3. get the missing information in the correct format,
4. add the missing information to the `robins` data.

The following commands identifies the missing commands. After each command, type the name of the object which has been created to see what each command has done.

```
robin.lab <- unique(robins$Sample.Label)
miss.lab <- tran.lab[!is.element(el = tran.lab,
    set = robin.lab)]
```

To understand what this latter command has done, it can be broken down into several elements:

- elements of `tran.lab` are selected using `[]`
- the `is.element` function (without the `!` symbol) selects the elements in `tran.lab`, which are also in the `set` argument (i.e. `robin.lab`)
- the `!` is used to select the elements in `tran.lab` that are NOT in `robin.lab`.

```
## Robins were detected on the following transects:
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10 12 13 14
## [14] 15 17 18
```

```
##
##  Therefore missing transects are:
```

```
## [1] 11 16 19
```

Now we know which transects are missing, we can select these records from the `birds` data frame:

```
miss.data <- birds[is.element(birds$Sample.Label,
    miss.lab), ]
```

However, the information about the transects are repeated in this new data frame because we have just selected all records for these transects. A quick check of the number of rows will confirm this:

```
length(miss.data$Sample.Label)
```

To get rid of rows where `Sample.Label` is duplicated use the command:

```
miss.data <- miss.data[!duplicated(miss.data$Sample.Label),
    ]
```

This command has selected the records from `miss.data` for which the transect label is not duplicated.

We only want to keep the information about search effort and so data in the `distance`, `species` and `visit` columns are set to missing:

```
miss.data$distance <- rep(NA, 3)
miss.data$species <- rep("NA", 3)
miss.data$visit <- rep(NA, 3)
```

Examine `miss.data`.

```
miss.data
```

```
##     Region.Label Area repeats Sample.Label
## 234     Montrave 33.2       2           11
## 299     Montrave 33.2       2           16
## 339     Montrave 33.2       2           19
##     Effort distance species visit
## 234  0.078       NA      NA    NA
## 299  0.378       NA      NA    NA
## 339  0.040       NA      NA    NA
```

The final thing to do is to add the missing data (`miss.data`) to the `robins` data frame using the `rbind` function (this combines data frames with the same columns).

```
robins <- rbind(robins, miss.data)
```

Let's see the result of all this manipulation:

```
tail(robins, n = 4)
```

```
##     Region.Label Area repeats Sample.Label
## 334     Montrave 33.2       2           18
## 234     Montrave 33.2       2           11
## 299     Montrave 33.2       2           16
## 339     Montrave 33.2       2           19
```

```
##      Effort distance species visit
## 334  0.400       70       r     2
## 234  0.078       NA      NA    NA
## 299  0.378       NA      NA    NA
## 339  0.040       NA      NA    NA
```

If we wanted to be very tidy, then the data frame could be sorted so that the transect labels were in order:

```
robins <- robins[order(robins$Sample.Label), ]
```

1.4 Analysis

Before we fit any models, have a quick look at the histogram of distances:

```
hist(robins$distance, breaks = 20)
```

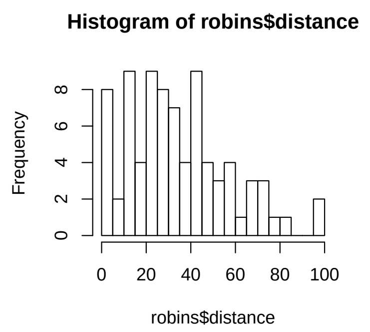


Figure 1: Perpendicular distances of robins in Montrave study.

Consistent with Buckland et al. (2015), three detection functions are fitted:

```
robin.hn.herm <- ds(robins, truncation = 95, transect = "line",
    key = "hn", adjustment = "herm", convert.units = 0.1)
robin.uni.cos <- ds(robins, truncation = 95, transect = "line",
    key = "unif", adjustment = "cos", convert.units = 0.1)
robin.haz.simp <- ds(robins, truncation = 95,
    transect = "line", key = "hr", adjustment = "poly",
    convert.units = 0.1)
```

Question: What is the preferred model for the robin data?

```
summarize_ds_models(robin.hn.herm, robin.uni.cos,
    robin.haz.simp)
```

Table 2: Model selection for robin data from Montrave line transect survey.

	Model	C-vM p-value	$\hat{P}_a$	se($\hat{P}_a$)	ΔAIC
2	`robin.uni.cos`	0.5098331	0.6356733	0.1028502	0.0000000
1	`robin.hn.herm`	0.3784802	0.5998042	0.0677143	0.3332786
3	`robin.haz.simp`	0.7316297	0.6790915	0.0527646	0.5649923

Note: All three detection function fit the data (based upon the C-vM test of exact distances). The estimated detection probability is very similar for all models, and the ΔAIC values of all models is < 1. Hence all models will give very similar estimates of density.

1.5 Examining the `dht` object

The fitted model object (e.g. `robin.uni.cos`) is made up of two parts; the detection function in the `ddf` part and the estimates in the `dht` part. In this section, we look at the `dht` part.

To list the elements that are contained in `dht`, use the `names` function:

```
names(robin.uni.cos$dht)
```

```
## [1] "individuals"
```

Detections were of individual birds and so group size was not included in these data - if it had been included (in a column called `size`), then as well as `individuals` there would have been elements `clusters` and `Expected.S`.

The estimates stored in the `individuals` object can be listed in a similar manner:

```
names(robin.uni.cos$dht$individuals)
```

```
## [1] "bysample"        "summary"
## [3] "N"               "D"
## [5] "average.p"       "cormat"
## [7] "vc"              "Nhat.by.sample"
```

To collect together the density estimates (and estimates of precision) from all the fitted models, we can use the following command:

```
# Collect together results
model.results <- rbind(robin.uni.cos$dht$individuals$D,
    robin.haz.simp$dht$individuals$D, robin.hn.herm$dht$individuals$D)
```

Question: Examine the three sets of density estimates to see if the previous suggestion (that the density estimates are similar) is confirmed.

```
model.results
```

```
##   Label  Estimate         se        cv
## 1 Total 0.6856860 0.13163867 0.1919810
## 2 Total 0.6418461 0.08298309 0.1292881
## 3 Total 0.7266910 0.11121789 0.1530470
##         lcl       ucl       df
## 1 0.4698684 1.0006319 89.83758
## 2 0.4948932 0.8324351 41.07649
## 3 0.5362652 0.9847362 65.18160
```

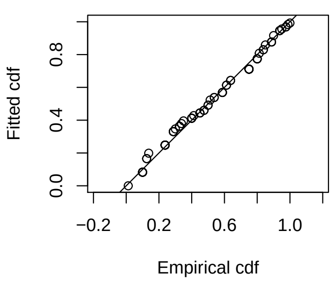
1.6 Goodness of fit

Here we look at goodness of fit test with unequal bin intervals and just consider one of the fitted models. First we specify the required bin intervals.

```
robin.brks <- c(0, 12.5, 22.5, 32.5, 42.5, 52.5,
    62.5, 77.5, 95)
```

Perform the tests using both exact distance data for the Cramer-von Mises test and specified breakpoints for χ^2 test for the uniform-cosine model that had the (slightly) smallest AIC score.

```
gof_ds(robin.uni.cos, breaks = robin.brks, chisq = TRUE)
```



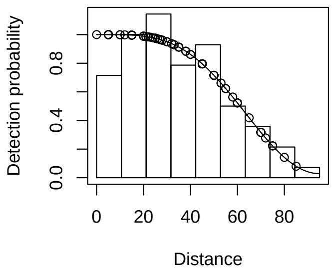
```
##
## Goodness of fit results for ddf object
##
## Chi-square tests
##              [0,12.5] (12.5,22.5] (22.5,32.5]
## Observed  11.000000  15.0000000  15.0000000
## Expected  16.553631  13.1496303  12.7439916
## Chisquare  1.863206   0.2603775   0.3993705
##             (32.5,42.5] (42.5,52.5]
## Observed   10.0000000  13.0000000
## Expected   11.7483337  10.0013400
## Chisquare   0.2601791   0.8990757
##             (52.5,62.5] (62.5,77.5]
## Observed   7.00000000  7.00000000
## Expected   7.58790911  6.35135600
## Chisquare  0.04555104  0.06624397
##              (77.5,95]      Total
## Observed  2.000000000 80.000000
## Expected  1.863807905 80.000000
## Chisquare 0.009951823  3.803955
##
## P = 0.57797 with 5 degrees of freedom
##
## Distance sampling Cramer-von Mises test (unweighted)
## Test statistic = 0.116511 p-value = 0.509833
```

Note: The detections fall close to the diagonal line of the qq plot, suggesting an adquate fit for the uniform cosine model. The *p-value* of the Cramer-von Mises test (at bottom of printout) confirms this. Similarly the *p-value* for the χ^2 test also suggests an adequate fit.

1.7 Customising the detection function plot

The `plot` function provides a basic plot of the fitted detection function overlaid onto the scaled distribution of distances:

```
plot(robin.uni.cos)
```



However, the plot can be customised for reporting:

```
plot(robin.uni.cos, showpoints = FALSE, black.white = TRUE,
    pl.den = 50, lwd = 2, breaks = robin.brks,
    main = "Uniform-cosine", xlab = "Distance (m)")
```

The arguments are:

- `showpoints` - logical indicating whether observed distances are shown
- `lwd` - line width (1=default)
- `pl.den` - density of shading of histogram (0=no shading)

For other options see `help(plot.ds)` (Note `plot` is a generic function which selects a relevant type of plot based the object).

References

Buckland, S. T. 2006. Point transect surveys for songbirds: robust methodologies. The Auk, **123**:345.

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.

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/.