Introduction to distance sampling

Workshop, 21-23 August 2019

Centre for Research into Ecological and Environmental Modelling

Exercise 3 Assessing line transect detection functions

In Exercise 2, we fitted different detection functions and compared them in terms of goodness of fit and AIC. Here, we continue to fit and assess different models and look at additional arguments in the `ds` package.

1 Objectives

The aim of this exercise is to practise fitting and assessing different line transect detection functions and in particular to:

1. Understand the data format when there are no detections on a line,
2. Explore different truncation options,
3. Determine whether adjustment terms are required,
4. Practice model selection,

2 Fitting models to simulated data

The data used for this practical were generated from a half-normal distribution and therefore the true density is known. There are 12 transects.

The data are stored in a file called 'IntroDS_3.1.csv' (in the `data` directory) which contains the following columns:

- Study.Area - Name of study called (not very imaginatively) 'LTExercise3'
- Region.Label - identifier of regions (in this case there is only one region and it is set to 'Default')
- Area - size of the study region (km^2)
- Sample.Label - line transect identifier (Line 1 - Line 12)
- Effort - length of the line transects (km)
- object - unique identifier to each detected object
- distance - perpendicular distances (metres).

2.1 Importing the data

If you have started a new `R` session, then you will need to load the `Distance` package again, otherwise, you can skip this command and move straight to importing the new data file.

```
# Load library (if not already loaded)
library(Distance)
# Import data
```

```
ltsimdata <- system.file("extdata", "IntroDS_3.1.csv", package = "dsdata")
ltdat <- read.csv(ltsimdata, header=TRUE)
# Check that it has been imported correctly
head(ltdat, n=3)
```

2.1.1 Data format: no detections on a transect

Before fitting models, it is worth investigating the data a bit further: let's start by summarising the perpendicular distances:

```
# Summary of perpendicular distances
summary(ltdat$distance)
```

The summary indicates that the minimum distance is `min(hndat$distance)` and the maximum is `max(hndat$distance)` metres and there is one missing value (indicated by the `NA`). If we print a few rows of the data, we can see that this missing value occurred on transect 'Line 11'.

```
# Print out a few lines of data
ltdat[100:102, ]
```

The `NA` indicates that there were no detections on this transect, but the transect information needs to be included otherwise the number of transects and the total line length will be incorrect.

2.2 Truncation

Let's start by fitting a basic model, i.e. no adjustment terms (by default a half normal model is fitted).

For this project, perpendicular distances are in metres and the transect lines are recorded in kilometres.

```
conversion.factor <- convert_units("meter", "kilometer", "square kilometer")
# Fit half normal, no adjustments
lt.hn <- ds(data=ltdat, key="hn", adjustment=NULL,
            convert.units=conversion.factor)
```

Looking at a summary of the model object, how many objects are there in total? What is the maximum observed perpendicular distance?

```
# Print a summary of the fitted detection function
summary(lt.hn)
```

Plot the detection function and specify many histogram bins:

```
plot(lt.hn, nc=30)
```

The histogram indicates that there is a large gap in detections therefore, to avoid a long right hand tail in the detection function, truncation is necessary. There are several ways to truncate: excluding distances beyond some specified distance or excluding a specified

percentage of the largest distances. Note that here we only consider excluding large perpendicular distances, which is frequently referred to as right truncation.

2.2.1 Truncation at a fixed distance

The following command truncates the perpendicular distances at 20 metres i.e. objects detected beyond 20 m are excluded.

```
# Truncate at 20metres
lt.hn.t20m <- ds(data=ltdat, key="hn", adjustment=NULL, truncation=20,
                 convert.units=conversion.factor)
```

Generate a summary and plot of the detection function to see what effect this truncation has had on the number of objects.

2.2.2 Truncating a percentage of distances

An alternative way to truncate distances is to specify a percentage of detected objects that should be excluded. In the command below, 10% of the largest distances are excluded.

```
# Truncate largest 10% of distances
lt.hn.t10per <- ds(data=ltdat, key="hn", adjustment=NULL, truncation="10%",
                   convert.units=conversion.factor)
```

Again, generate a summary and plot to see what effect this has had.

```
summary(lt.hn.t10per)
plot(lt.hn.t10per)
```

2.3 Exploring different models

Decide on a suitable truncation distance (but don't spend too long on this) and then fit different key detection functions and adjustment terms to assess whether these data can be satisfactorily analysed with the 'wrong' model. By default, the `ds` function fits a half normal function and cosine adjustment terms (`adjustment="cos"`) of up to order 5: AIC is used to determine how many, if any, adjustment terms are required. This model is specified in the command below:

```
# Half normal detection, cosine adjustments, no truncation
lt.hn.cos <- ds(data=ltdat, key="hn", adjustment="cos",
                convert.units=conversion.factor)
```

Change the key and adjustment terms: possible options are listed in Exercise 2 or use the `help(ds)` for options. See how the bias and precision compare between the models: the true density was 79.8 animals per km^2.

3 Fitting models to real data (optional question)

- The objective of this portion of the exercise is to give you more practice with a line transect data set and also familiarise you with converting from exact distances to binned distances.

The data stored in a text file called 'IntroDS_3.2.csv' were collected during a line transect survey of capercaillie (a species of large grouse) in Scotland. The data consist of:

- Region.Label - name of the region
- Area - size of region (hectares[1])
- Sample.Label - line transect identifier (one transect)
- Effort - length of line (km)
- object - unique identifier for each group detected
- distance - perpendicular distance to each group (metres)
- size - group size (in this case only single birds were detected)

Import these data and:

1. Decide on a suitable truncation distance, if any,
2. Fit a few different key detection functions and adjustment terms
3. Compare the AIC values and qq plots and choose a model
4. Calculate density for your chosen model. To obtain density in birds per hectare, the conversion units should be specified as `convert.units=0.1`.

3.1 Converting exact distances to binned distances

Sometimes we wish to convert exact distances to binned distances, if for example, there is evidence of rounding to favoured values. To do this in `ds` we need to specify the cutpoints of the bins, including zero and the maximum distance. In the example below, cutpoints at 0, 10, 20, ..., 80 are specified.

```
# Import data
capercaillefile <- system.file("extdata", "IntroDS_3.2.csv", package = "dsdata")
caper <- read.csv(capercaillefile, header=TRUE)
# Specify cutpoint for bins
bins <- seq(from=0, to=80, by=10)
conversion.factor <- convert_units("meter", "kilometer", "hectare")
# Specify model with binned distances
caper.bin <- ds(data=caper, key="hn", cutpoints=bins,
                convert.units=conversion.factor)
# Plot
plot(caper.bin)
```

[1] 1 hectare = 100m × 100m

Solution 3. Assessing line transect detection functions

4 Fitting models to simulated data

```
# Load library
library(Distance)
# Import data
ltsimdata <- system.file("extdata", "IntroDS_3.1.csv", package = "dsdata")
ltdat <- read.csv(ltsimdata, header=TRUE)
# Check that it has been imported correctly
head(ltdat, n=3)
```

```
##    Study.Area Region.Label Area Sample.Label Effort object distance
## 1 LTExercise3      Default    1       Line 1      5      1      7.9
## 2 LTExercise3      Default    1       Line 1      5      2     10.2
## 3 LTExercise3      Default    1       Line 1      5      3     12.4
```

```
# How many observations (remember no detections on line 11)
max(ltdat$object, na.rm=TRUE)
```

```
## [1] 105
```

These data contain 105 observations. There were no detections on Line 11 and the format below indicates that `NA` is used to specify this.

```
ltdat[100:102, ]
```

```
##      Study.Area Region.Label Area Sample.Label Effort object distance
## 100 LTExercise3      Default    1      Line 10      7    100     16.6
## 101 LTExercise3      Default    1      Line 11      3     NA       NA
## 102 LTExercise3      Default    1      Line 12      4    101      1.0
```

Here we can see the effect of the different truncation options.

```
conversion.factor <- convert_units("meter", "kilometer", "square kilometer")
# Truncate at 20metres
lt.hn.t20m <- ds(data=ltdat, key="hn", adjustment=NULL, truncation=20,
                 convert.units=conversion.factor)
summary(lt.hn.t20m)
```

```
##
## Summary for distance analysis
## Number of observations :  103
## Distance range         :  0  -  20
##
## Model : Half-normal key function
## AIC   : 599.4236
##
```

```
## Detection function parameters
## Scale coefficient(s):
##             estimate        se
## (Intercept) 2.388955 0.1176774
##
##                         Estimate          SE         CV
## Average p              0.6377021  0.05317065 0.08337851
## N in covered region 161.5174243 16.52650082 0.10232024
##
## Summary statistics:
##    Region Area CoveredArea Effort   n  k       ER   se.ER     cv.ER
## 1 Default    1        1.92     48 103 12 2.145833 0.32184 0.1499837
##
## Abundance:
##   Label Estimate       se        cv      lcl      ucl       df
## 1 Total 84.12366 14.43575 0.1716015 58.86701 120.2166 18.65739
##
## Density:
##   Label Estimate       se        cv      lcl      ucl       df
## 1 Total 84.12366 14.43575 0.1716015 58.86701 120.2166 18.65739
```

This has excluded 2 observations.

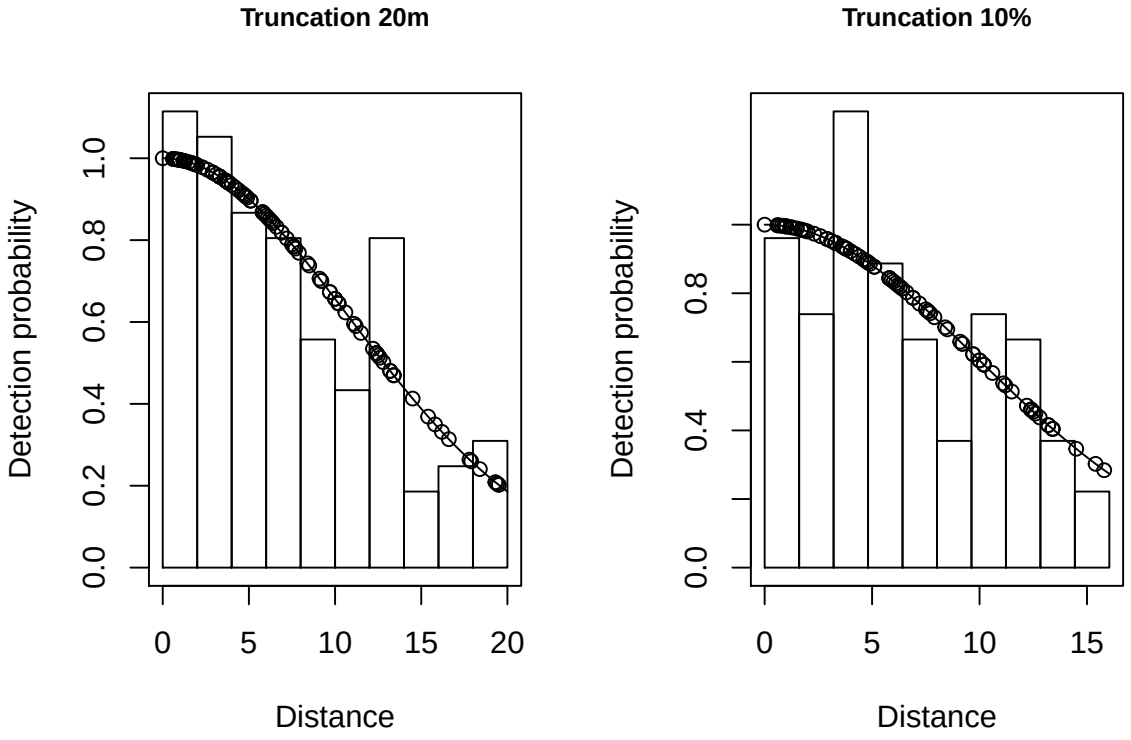
```
# Truncate 10% of largest distances
lt.hn.t10per <- ds(data=ltdat, key="hn", adjustment=NULL, truncation="10%",
                   convert.units=conversion.factor)
summary(lt.hn.t10per)
```

```
##
## Summary for distance analysis
## Number of observations :  94
## Distance range         :  0  -  16.04
##
## Model : Half-normal key function
## AIC   : 512.2441
##
## Detection function parameters
## Scale coefficient(s):
##             estimate        se
## (Intercept) 2.298481 0.1606186
##
##                         Estimate          SE         CV
## Average p              0.6946953  0.06767649 0.09741895
## N in covered region 135.3111280 15.27181080 0.11286441
##
## Summary statistics:
##    Region Area CoveredArea Effort  n  k       ER    se.ER     cv.ER
## 1 Default    1     1.53984     48 94 12 1.958333 0.278492 0.1422087
##
```

```
## Abundance:
##   Label Estimate       se        cv      lcl      ucl       df
## 1 Total  87.8735 15.14735 0.1723768 61.68485 125.1807 23.14385
##
## Density:
##   Label Estimate       se        cv      lcl      ucl       df
## 1 Total  87.8735 15.14735 0.1723768 61.68485 125.1807 23.14385
```

This has excluded 11 observations. The plots are shown below.

```
# Divide plot window
par(mfrow=c(1,2))
plot(lt.hn.t20m, main="Truncation 20m")
plot(lt.hn.t10per, main="Truncation 10%")
```



A few different models are shown below.

```
# Fit a few different models
# Half normal model, no adjustments, no truncation
lt.hn <- ds(data=ltdat, key="hn", adjustment=NULL, convert.units=conversion.factor)
# Half normal model, cosine adjustments, truncation at 20m
lt.hn.cos.t20m <- ds(data=ltdat, key="hn", adjustment="cos", truncation=20,
                     convert.units=conversion.factor)
# Uniform model, cosine adjustments, truncation at 20m
lt.uf.cos.t20m <- ds(data=ltdat, key="unif", adjustment="cos",
                     truncation=20, convert.units=conversion.factor)
# Hazard rate model, no adjustments, truncation at 20m
lt.hr.t20m <- ds(data=ltdat, key="hr", adjustment="poly", truncation=20,
```

```
                convert.units=conversion.factor)
```

The results are shown in the table below: 'Terms' indicates the number of selected adjustment terms and 'Pa' is the estimated detection probability.

Table 1: Results for simulated data with differing truncation and detection functions. (continued below)

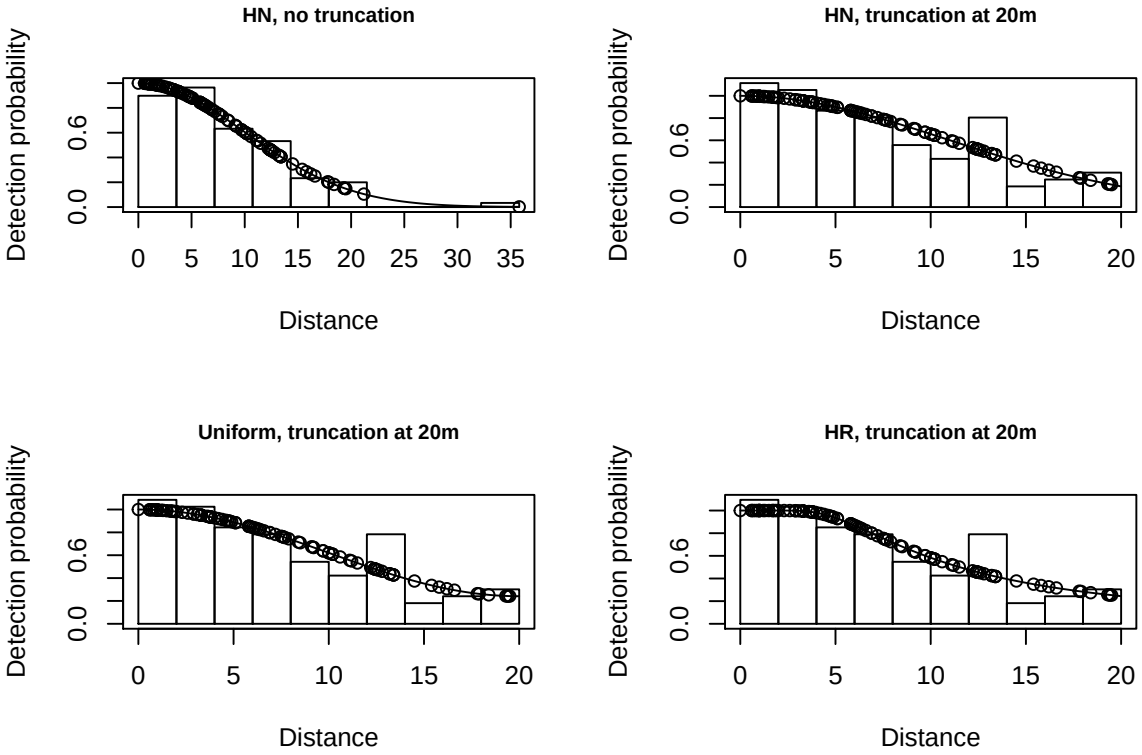
DetectionFunction	Adjustments	Terms	Truncation	AIC	Pa
Half-nomal	None	0	35.8	636.9	0.3492
Half-nomal	Cosine	0	20	599.4	0.6377
Uniform	Cosine	1	20	598.6	0.6207
Hazard rate	Polynomial	0	20	600.7	0.6263

Density	D.CV	Lower.CI	Upper.CI
87.49	0.1576	62.7	122.1
84.12	0.1716	58.87	120.2
86.43	0.1663	60.96	122.5
85.65	0.2034	56.9	128.9

There is a change in $\hat{P}_a$ due to truncation but all the models provide very similar density results, although precision is slightly larger for the hazard rate model (because more parameters are estimated). Agreement between the estimate and the known true density is less good if you do not truncate the data, or do not truncate sufficiently. Note that the AIC values can only be compared for models with the same truncation and hence the same objects.

The take home message is that, with care, we can get reliable estimates using the wrong model (remember the data were simulated using a half normal detection function): this is useful because, in practise, the 'correct' model is never known.

```
# Divide plot window
par(mfrow=c(2,2))
# Plot detection functions
plot(lt.hn, main="HN, no truncation")
plot(lt.hn.cos.t20m, main="HN, truncation at 20m")
plot(lt.uf.cos.t20m, main="Uniform, truncation at 20m")
plot(lt.hr.t20m, main="HR, truncation at 20m")
```



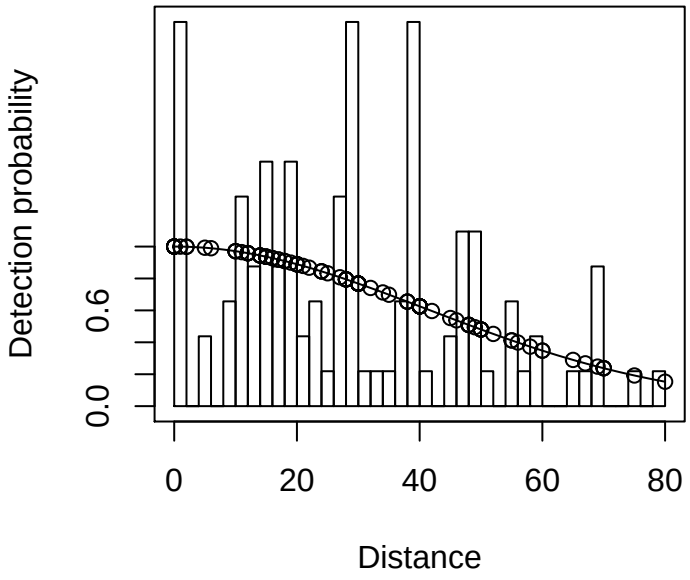
5 Fitting models to real data (optional)

After importing these data, a basic model is fitted and plotted to determine if truncation is required.

```
# Import data
capercaillefile <- system.file("extdata", "IntroDS_3.2.csv", package = "dsdata")
caper <- read.csv(capercaillefile, header=TRUE)
# Check data OK
head(caper, n=3)
```

```
##       Region.Label Area Sample.Label Effort object distance size
## 1 Monaughty Forest 1472            1    240      1       28    1
## 2 Monaughty Forest 1472            1    240      2       17    1
## 3 Monaughty Forest 1472            1    240      3       15    1
```

```
conversion.factor <- convert_units("meter", "kilometer", "hectare")
# Fit a half normal model with no adjustments and no truncation
caper.hn <- ds(data=caper, key="hn", adjustment=NULL,
               convert.units=conversion.factor)
# Plot with lots of bins, each of width 2m
plot(caper.hn, nc=40)
```



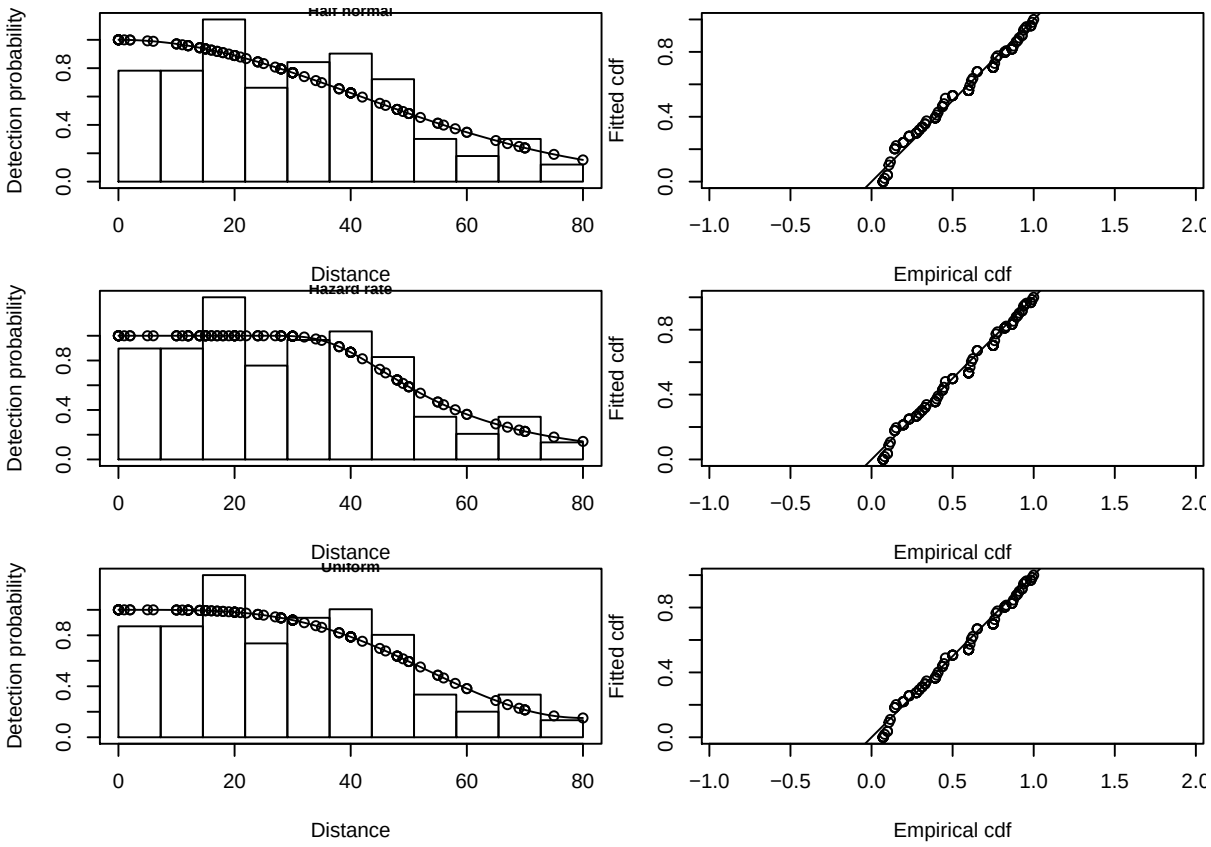
There isn't a long tail to the detection function and so no truncation will be used.

There may be evidence of rounding to some values (e.g. 0, 30, 40, 70) however, we will ignore this at present (but address it below) and fit the three alternative key functions and use the default setting for adjustments terms (i.e. cosine up to order 5).

```
# Fit different models allowing cosine adjustments if required
# Half normal model
caper.hn.cos <- ds(data=caper, key="hn", adjustment="cos",
                   convert.units=conversion.factor)
# Hazard rate model
caper.hr.cos <- ds(data=caper, key="hr", adjustment="cos",
                   convert.units=conversion.factor)
# Uniform model
caper.uf.cos <- ds(data=caper, key="unif", adjustment="cos",
                   convert.units=conversion.factor)
```

The detection functions and qq plots are shown below:

```
# Divide plot window
par(mfrow=c(3,2))
par(mar=c(4,4,.2,.1))
plot(caper.hn.cos, main="Half normal")
gof_ds(caper.hn.cos)
plot(caper.hr.cos, main="Hazard rate")
gof_ds(caper.hr.cos)
plot(caper.uf.cos, main="Uniform")
gof_ds(caper.uf.cos)
```



Summarise the goodness of fit statistics (in a pretty format). This table indicates that the hazard rate detection function had the lowest AIC but the difference in AIC between all three models was small.

```
pander::pander(summarize_ds_models(caper.hn.cos, caper.hr.cos, caper.uf.cos),
               caption="Summary of results of Capercaillie analysis.")
```

Table 3: Summary of results of Capercaillie analysis. (continued below)

	Model	Key function	Formula
2	`caper.hr.cos`	Hazard-rate	~1
1	`caper.hn.cos`	Half-normal	~1
3	`caper.uf.cos`	Uniform with cosine adjustment terms of order 1,2	NA

	C-vM p-value	$\hat{P}_a$	se($\hat{P}_a$)	ΔAIC
2	0.6625	0.7026	0.05249	0
1	0.3322	0.6128	0.05303	0.03135
3	0.6129	0.6818	0.09784	0.2803

The results for the three different models are shown below: density is in birds per ha.

Table 5: Capercaillie point estimates of density and associated measures of precision.

DetectionFunction	AIC	Pa	Density	D.CV	Lower.CI	Upper.CI
Half-nomal	957.9	0.6128	0.04759	0.1475	0.02732	0.08291
Hazard rate	957.9	0.7026	0.04151	0.1481	0.02045	0.08426
Uniform	958.2	0.6818	0.04278	0.191	0.02644	0.06921

These capercaillie data are reasonably well-behaved and different models that fit the data well should give similar results.

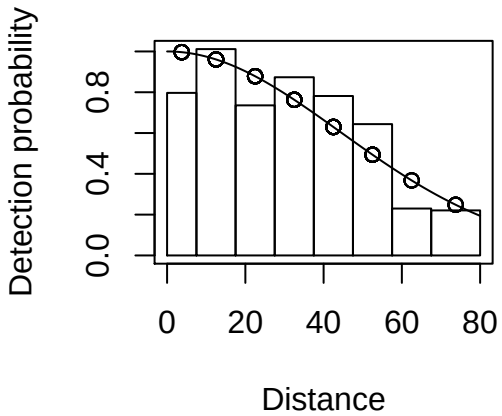
5.1 Converting exact distances to binned distances

To deal with rounding in the distance data, the exact distances can be converted into binned distances. The cutpoints need to be chosen with care so that the distance bins are sufficiently wide enough to ensure that the 'correct' perpendicular distance is in the band containing the rounded recorded value. The bin widths do not have to be equal, as shown in example here: the cutpoints are 0, 7.5, 17.5, 27.5, ..., 67.5, 80.0 m. Note, that any distances beyond the largest bin will be excluded.

```
# Specify (uneven) cutpoint for bins
bins <- c(0, seq(from=7.5, to=67.5, by=10), 80)
# Check bins
bins
```

```
## [1]  0.0  7.5 17.5 27.5 37.5 47.5 57.5 67.5 80.0
```

```
# Specify model with binned distances
caper.hn.bin <- ds(data=caper, key="hn", adjustment="cos", cutpoints=bins,
                   convert.units=conversion.factor)
# Plot
plot(caper.hn.bin)
```



```
# Summarise results
caper.hn.bin$dht$individuals$summary
```

```
##             Region Area CoveredArea Effort   n        ER se.ER cv.ER
## 1 Monaughty Forest 1472        3840    240 112 0.4666667     0     0
##   mean.size se.mean
## 1         1       0
```

```
caper.hn.bin$dht$individuals$D[1:6]
```

```
##   Label   Estimate          se        cv        lcl       ucl
## 1 Total 0.04531495 0.006899613 0.1522591 0.02587617 0.0793566
```

Note that the binning of the data results in virtually identical estimates of density (0.045 birds per ha) and essentially no change in the precision of the density estimate compared with the estimates with analysis of exact distance data.