Rdistance Tutorial for Beginners

Jason D. Carlisle and Trent L. McDonald

April 22, 2015

This tutorial is meant to be a beginner's guide to using `Rdistance`. It is assumed that you have some familiarity with using Program R, but not necessarily with distance-sampling analysis. This beginning tutorial focuses on input data requirements, fitting a detection function, and estimating abundance (or density). Here, we make use of the example datasets already contained within `Rdistance` (i.e., line transect surveys of sparrows), so you can complete this tutorial without having any data of your own. This tutorial is current as of version 1.2.2 of `Rdistance`.

1: Install and load Rdistance

If you haven't already done so, install the latest version of `Rdistance`. In the R console, issue `install.packages("Rdistance")`. After the package is installed, it can be loaded into the current session as follows:

```
require(Rdistance)
```

```
## Loading required package: Rdistance
## Rdistance (version 1.2.2)
```

2: Read in input data

`Rdistance` requires two input datasets. These can be prepared outside of `R` and read in as data.frames using, for example, `read.csv`. In the following sections, we make use of the sparrow example datasets already contained within `Rdistance`.

The first required dataset is a detection data.frame, with a row for each detection, and the following required columns, named as follows:

- `siteID` = Factor, the site or transect where the detection was made.
- `groupsize` = Numeric, the number of individuals within the detected group.
- `dist` = Numeric, the perpendicular distance (also known as off-transect distance) from the transect to the detected group.

If the observers recorded sighting distance and sighting angle instead of perpendicular distance (as is often common in line transect surveys), you can use the `perp.dists` function (detailed in Section 3) to calculate the perpendicular distances based on the sighting distances and sighting angles.

The second required dataset is a transect data.frame, with a row for each transect surveyed, and the following required columns, named as follows:

- `siteID` = Factor, the site or transect surveyed.
- `length` = Numeric, the length of the transect. Use the same units as the detection distances.
- `...` = Any additional transect-level covariate columns.

3: Fit a detection function

After prepping the input data, the first step is to explore your data and fit a detection function. First, load the example dataset of sparrow detections and their distances from the package using `data(sparrow.detections)`. Be sure that you have installed and loaded `Rdistance` prior to issuing the following commands:

```r
data(sparrow.detections)
head(sparrow.detections)
```

```
##   siteID groupsize sightdist sightangle
## 1     A1         1        65         15
## 2     A1         1        70         10
## 3     A1         1        25         75
## 4     A1         1        40          5
## 5     A1         1        70         85
## 6     A1         1        10         90
```

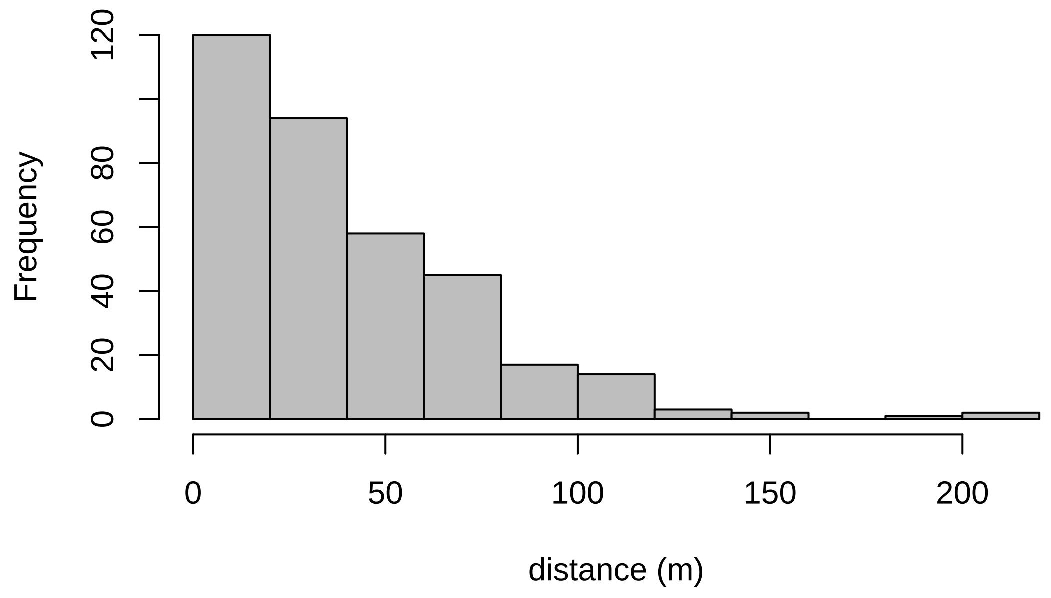
Note that the observers recorded sighting distances and sighting angles. Use the `perp.dists` function to calculate the perpendicular distance from each detected group to the transect, then remove the `sightdist` and `sightangle` columns.

```r
sparrow.detections$dist <- perp.dists(obs.dist=sparrow.detections$sightdist,
                                      obs.angle=sparrow.detections$sightangle)
sparrow.detections <- sparrow.detections[, -which(names(sparrow.detections)
                                                  %in% c("sightdist", "sightangle"))]
head(sparrow.detections)
```

```
##   siteID groupsize dist
## 1     A1         1 16.8
## 2     A1         1 12.2
## 3     A1         1 24.1
## 4     A1         1  3.5
## 5     A1         1 69.7
## 6     A1         1 10.0
```

Explore the distribution of distances.

```r
hist(sparrow.detections$dist, col="grey", main="", xlab="distance (m)")
```

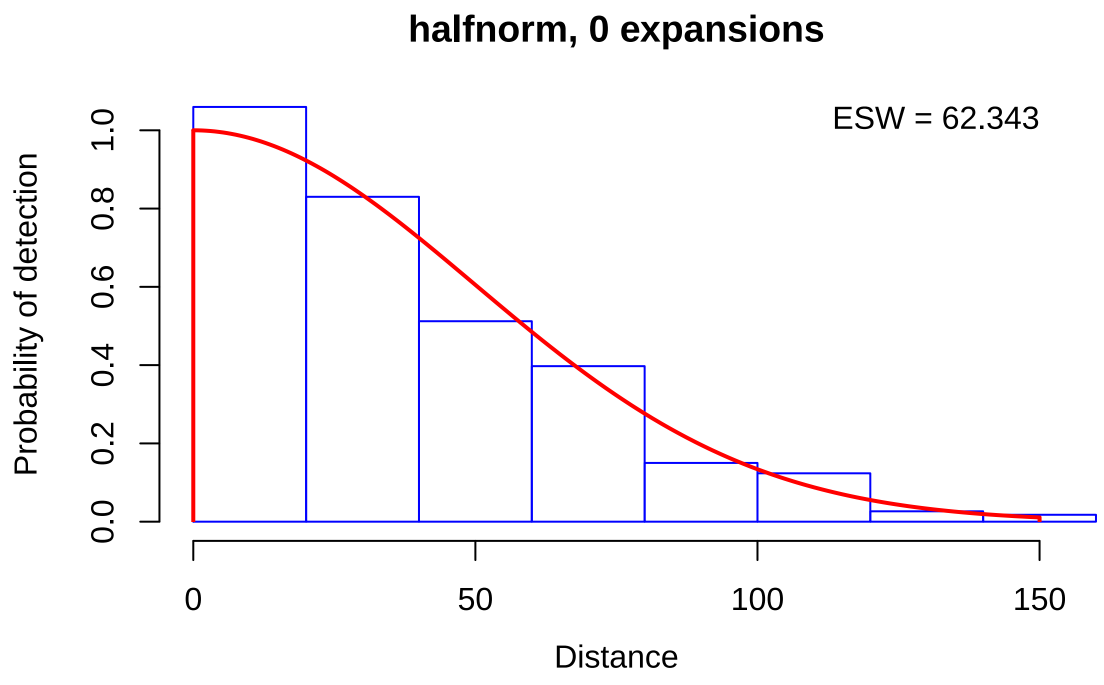


```
summary(sparrow.detections$dist)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   14.15   30.75   39.64   57.35  207.00
```

Next, fit a detection function (plotted as a red line) using `F.dfunc.estim`. For now, we will proceed using the half-normal likelihood as the detection function, but in Section 5 of this tutorial, we demonstrate how to run an automated process that fits multiple detection functions and compares them using AICc. Note that distances greater than 150 m are quite sparse, so here we right-truncate the data, tossing out detections where `dist` $>$ 150.

```
dfunc <- F.dfunc.estim(sparrow.detections, likelihood="halfnorm", w.hi=150)
plot(dfunc)
```



```
dfunc
```

```
## Call: F.dfunc.estim(dist = sparrow.detections, likelihood = "halfnorm",     w.hi = 150)
##
## Coefficients:
##    Sigma
## 49.87415
##
## Convergence: Success
## Function: HALFNORM
## Strip: 0 to 150
## Effective strip width: 62.34334
## Scaling: g(0) = 1
## Log likelihood: 1630.716
## AIC: 3263.443
```

The effective strip width (ESW) is the key information from the detection function that will be used to next estimate abundance (or density). The ESW is calculated by integrating under the detection function. A survey with imperfect detection and ESW equal to X effectively covers the same area as a study with perfect detection out to a distance of X. See the help documentation for `ESW` for details.

4: Estimate abundance given the detection function

Estimating abundance requires the additional information contained in the second required dataset, described earlier, where each row represents one transect. Load the example dataset of surveyed sparrow transects from the package.

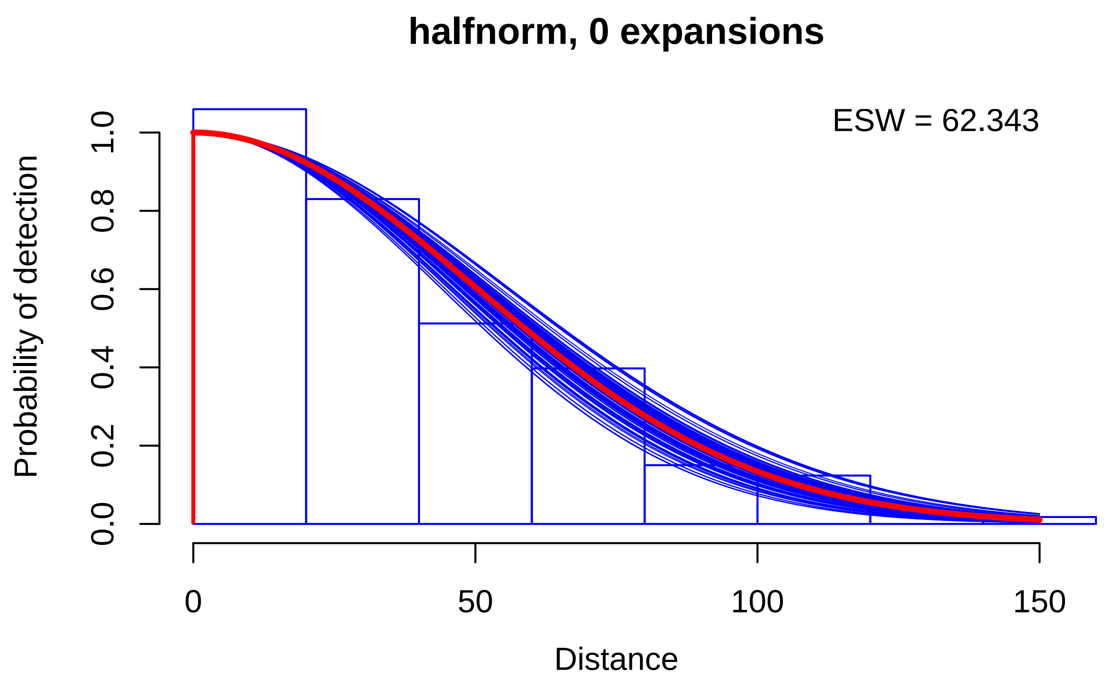
```r
data(sparrow.transects)
head(sparrow.transects)
```

```
##   siteID length sage
## 1     A1    500 High
## 2     A2    500 High
## 3     A3    500 High
## 4     A4    500 High
## 5     B1    500 High
## 6     B2    500 High
```

Next, estimate abundance (or density in this case) using `F.abund.estim`. If `area`=1, then density is given in the squared units of the distance measurements – in this case, sparrows per square meter. Instead, we set `area`=10000 in order to convert to sparrows per hectare (1 ha == 10,000 m^2). The equation used to calculate the abundance estimate is detailed in the help documentation for `F.abund.estim`.

Confidence intervals for abundance are calculated using a bias-corrected bootstrapping method (see `F.abund.estim`), and the detection function fit in each iteration of the bootstrap is plotted as a blue line (if `plot.bs=TRUE`). Note that, as with all bootstrapping procedures, there may be slight differences in the confidence intervals between runs due to so-called 'simulation slop'. Increasing the number of bootstrap iterations (`R` = 100 used here) may be necessary to stabilize CI estimates.

```r
fit <- F.abund.estim(dfunc, detection.data=sparrow.detections,
                     transect.data=sparrow.transects,
                     area=10000, R=100, ci=0.95, plot.bs=TRUE)
```



```
## Computing bootstrap confidence interval on N...
## ===========================================================================
fit
```

```
## Call: F.dfunc.estim(dist = sparrow.detections, likelihood = "halfnorm",     w.hi = 150)
##
## Coefficients:
##    Sigma
## 49.87415
##
## Convergence: Success
## Function: HALFNORM
## Strip: 0 to 150
## Effective strip width: 62.34334
## Scaling: g(0) = 1
## Log likelihood: 1630.716
## AIC: 3263.443
##
## Abundance estimate:  0.8265162 ;  95% CI=( 0.6925541 to 1.04618 )
```

Results of interest (such as the abundance estimate and confidence interval) can be extracted from the resulting object (here called `fit`).

```
fit$n.hat
```

```
## [1] 0.8265162
```

```
fit$ci
```

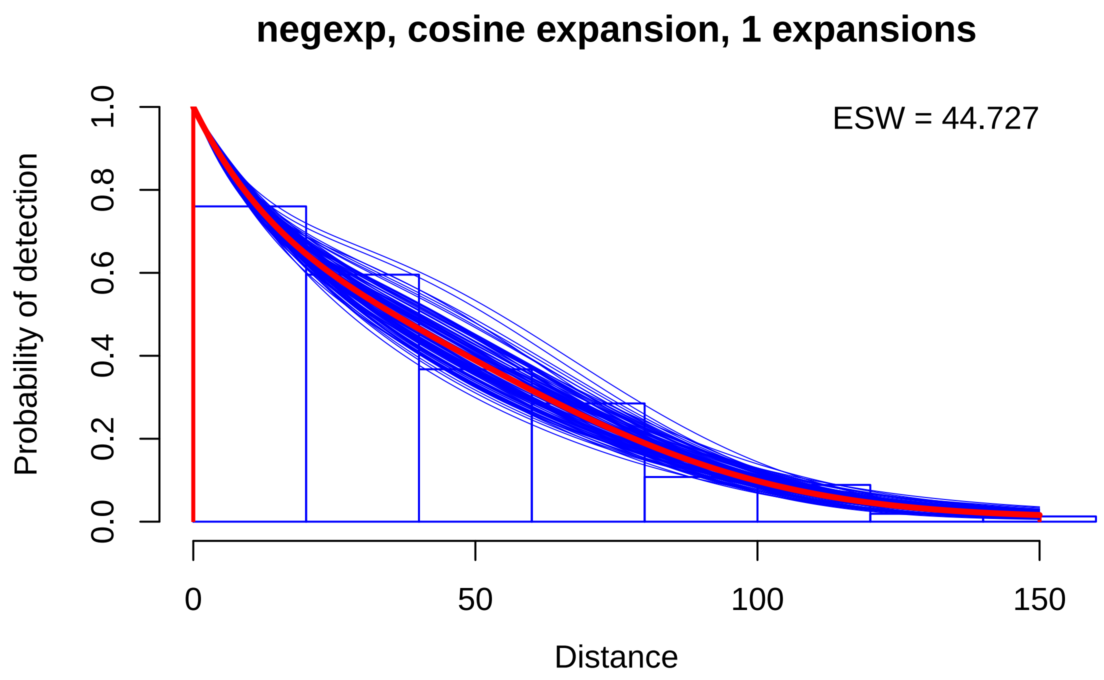
```
## 3.146705% 98.03137%
## 0.6925541 1.0461801
```

5: Use AICc to select a detection function and estimate abundance

Alternatively, steps 3 (fitting a detection function) and 4 (estimating abundance) can be automated using the function `F.automated.CDA`. This function attempts to fit multiple detection functions, uses AICc (by default, but see help documentation for `AIC.dfunc` for other options) to find the 'best' detection function, then proceeds to estimate abundance using that detection function. By default, `F.automated.CDA` tries dozens of detection functions, but you can restrict the process to fewer detection functions if you choose (see help documentation for `F.automated.CDA`). Specifying `plot=TRUE` would return a plot of each detection function. In this example, we won't restrict the number of detection functions attempted, and we won't plot each (`plot=FALSE`).

```
auto <- F.automated.CDA(detection.data=sparrow.detections,
                        transect.data=sparrow.transects,
                        w.hi=150, plot=FALSE, area=10000, R=100, ci=0.95, plot.bs=TRUE)
```

```
## Likelihood   Series  Expans  Converged?  Scale?  AIC
## halfnorm cosine  0   Yes     Ok  3263.4429
## halfnorm cosine  1   Yes     Ok  3261.585
## halfnorm hermite 1   Yes     Ok  3262.8066
## halfnorm simple  1   Yes     Ok  3262.1526
## halfnorm cosine  2   Yes     Ok  3263.3155
## halfnorm hermite 2   Yes     Ok  3263.593
## halfnorm simple  2   Yes     Ok  3263.4741
## halfnorm cosine  3   Yes     Ok  3265.1285
## halfnorm hermite 3   Yes     Ok  3265.5499
## halfnorm simple  3   Yes     Ok  3265.216
## hazrate      cosine  0   Yes     Ok  3267.6246
## hazrate      cosine  1   Yes     Ok  3263.3092
## hazrate      hermite 1   No      NA  NA
## hazrate      simple  1   No      NA  NA
## hazrate      cosine  2   Yes     Ok  3265.0704
## hazrate      hermite 2   Yes     Ok  3268.9682
## hazrate      simple  2   Yes     Ok  3265.5796
## hazrate      cosine  3   Yes     Ok  3267.117
## hazrate      hermite 3   Bad     NA  NA
## hazrate      simple  3   No      NA  NA
## uniform      cosine  0   Yes     Ok  3260.7318
## uniform      cosine  1   Bad     NA  NA
## uniform      hermite 1   Yes     Ok  3262.736
## uniform      simple  1   Yes     Ok  3262.2602
## uniform      cosine  2   Bad     NA  NA
## uniform      hermite 2   No      NA  NA
## uniform      simple  2   Bad     NA  NA
## uniform      cosine  3   Bad     NA  NA
## uniform      hermite 3   No      NA  NA
## uniform      simple  3   Bad     NA  NA
## negexp       cosine  0   Yes     Ok  3263.8153
## negexp       cosine  1   Yes     Ok  3260.0985
## negexp       hermite 1   Yes     Ok  3260.3108
## negexp       simple  1   Yes     Ok  3261.0097
## negexp       cosine  2   Yes     Ok  3262.0306
## negexp       hermite 2   No      NA  NA
## negexp       simple  2   Yes     Ok  3262.2878
## negexp       cosine  3   Yes     Ok  3263.4898
## negexp       hermite 3   No      NA  NA
## negexp       simple  3   Yes     Ok  3264.2055
## Gamma            0   Yes     Ok  3586.8919
## Note: Some models did not converge or had parameters at their boundaries.
```



```
## Computing bootstrap confidence interval on N...
## ===========================================================================
##
##
## ---------------- Final Automated CDS Abundance Estimate ------------------------------
## Call: F.dfunc.estim(dist = dist, likelihood = fit.table$like[1], w.lo = w.lo,     w.hi = w.hi, expans
##
## Coefficients:
##        Beta           a1
##  0.02754839  -0.26542176
##
## Convergence: Success
## Function: NEGEXP with 1 expansion(s) of COSINE series
## Strip: 0 to 150
## Effective strip width: 44.72749
## Scaling: g(0) = 1
## Log likelihood: 1628.032
## AIC: 3260.098
##
## Abundance estimate:  1.152038 ;  95% CI=( 0.9034703 to 1.502093 )
```

You can see that the detection function with the lowest AICc value (and thus selected as the 'best') is the negative exponential likelihood, with one cosine expansion.

6: Conclusion

Note that the detection function that you select has a large influence on the resulting abundance estimate. In sections 3 and 4, we fit a half-normal detection function and used that function to estimate sparrow density.

Our estimate was 0.83 sparrows per ha (95% CI = 0.69 - 1.05). In section 5, we used AICc to determine the best-fitting detection function and used that function to estimate sparrow density again. Our new estimate was 1.15 sparrows per ha (95% CI = 0.9 - 1.5). (Note, recall that your estimates may vary slightly from these due to minor 'simulation slop' inherent in bootstrapping methods). Thus we see that choosing an appropriate detection function is critical to accurately estimating abundance. The `F.automated.CDA` function can help you select a detection function that fits your data well.

That concludes this `Rdistance` tutorial. You are now ready to read in your own data, fit a detection function, and estimate abundance.