Beginner's Guide to Rdistance Line-Transect Analysis

No Covariates

Trent McDonald, Jason Carlisle, and Abigail Hoffman

2023-06-12

Introduction

This beginner's tutorial demonstrates estimation of density and abundance in `Rdistance`. We discuss data requirements, basic detection function estimation, and estimation of abundance (and density). This tutorial is for line transect surveys when detection does not depend on covariates.

1: Install and load Rdistance

First, install the latest version of `Rdistance`. In the R console, issue,

```
install.packages("Rdistance")
```

You can install the development version of `Rdistance` from GitHub using

```
devtools::install_github("tmcd82070/Rdistance")
```

After the package is installed, it is loaded into the current session with the `require` statement, i.e.,

```
require(Rdistance)
```

The following statements bring the example data sets into R's global environment where other routines can see them.

```
data("sparrowDetectionData")
data("sparrowSiteData")
```

2: Input data

Estimation of abundance (or density) in `Rdistance` requires two data sets; the *detection* data set and the *sites* data set. Both data sets can be prepared using any method that results in an R `data.frame`. For example, analysts can format detection and transect information on separate sheets in `Excel`, export each sheet to CSV files, and read them into R using `read.csv`. Importing data is beyond the scope of this vignette. Here, we use the Brewer's sparrow data set that comes bundled with `Rdistance`.

Detection Data

The *detection* data frame contains one row for each detected target, whether that target is a single individual or a group of individuals. Columns in the detection data frame contain information like group size and perpendicular off-transect distance to the group. At a minimum, `Rdistance` requires the following information in the *detection* data frame:

- **Detection Distance** = The perpendicular distance (also known as off-transect distance) from the transect to the detected group or individual. `Rdistance` requires that this column of the detection

data frame have measurement units (e.g., "m" or "ft"). Measurement units can be assigned using the `units()<-` or `units::set_units` functions. For example, `units(sparrowDetectionData$dist) <- "m"` makes all the measurements in column `dist` into meters. If column `dist` already has units, assigning meters in this way converts from the previous units into meters. Measurement units are required on all distances in `Rdistance` (e.g., off-transect distances, left and right cut-offs, scale locations, transect lengths, and study area size). Additional details are in `help(dfuncEstim)`.

- **Site ID** = The ID of the transect on which the detection was made. Transect IDs are required to estimate abundance (in function `abundEstim`), but not to estimate a distance function (in function `dfuncEstim`).
- **Group Sizes** = The number of individuals in the group associated with each detection. Group sizes are required to estimate abundance (function `abundEstim`), but not to estimate a distance function (function `dfuncEstim`). If all groups are size 1, group sizes can be omitted.

The specific columns that contain distances and group sizes can be named anything and are specified in the formula argument of function `dfuncEstim`. Likewise, the `siteID` column(s) can be named anything and is specified in the `transectID` argument to function `dfuncEstim`. If `siteID` is not specified (or is NULL), `Rdistance` constructs the `siteID`s from the set of common columns contained in the *detection* and *sites* data frames. See the **Input data frames** section of `help(dfuncEstim)` for additional details.

Line-transect distance-sampling analysis is performed on perpendicular off-transect distances, where off-transect distances are measured from the detected group to the closest point on the transect. In the field, observers commonly record straight-line sighting distance (from observer straight to the target) and sighting angle instead of perpendicular distance. `Rdistance` provides a utility function named `perpDists` that will compute perpendicular distances from sighting distance and angle. See `help(perpDists)` for details.

For reference, the first six rows of the sparrow *detection* data set are:

```
head(sparrowDetectionData)
```

```
##   siteID groupsize sightdist sightangle     dist
## 1     A1         1        65         15 16.8 [m]
## 2     A1         1        70         10 12.2 [m]
## 3     A1         1        25         75 24.1 [m]
## 4     A1         1        40          5  3.5 [m]
## 5     A1         1        70         85 69.7 [m]
## 6     A1         1        10         90 10.0 [m]
```

We will use `siteID`, `groupsize`, and `dist` in this tutorial. Details on the study and other columns are in `help(sparrowDetectionData)`.

Site Data

The *sites* data frame contains one row for each surveyed site regardless of whether the site was "positive" (one or more targets detected) or "zero" (no targets detected). We use the term 'site' because it encompasses both continuous transects and point transects. In this tutorial, a 'site' is one transect. `Rdistance` views the study area as containing one or more surveyed 'sites', where each 'site' has a particular surveyed length. Along the surveyed length, zero or more groups were detected. Stratum and sub-units of the study area are not automatically accommodated in `Rdistance`. We view stratum as separate study areas, estimate abundance in each separately, and then sum over stratum to arrive as the study area estimate.

At a minimum, `Rdistance` requires the following information in the *sites* data frame:

- **Site IDs** = The ID of every surveyed site (here, transect) in the study area.
- **Length** = The length of transect surveyed in each site. The length column in the *sites* data frame must have measurement units assigned (e.g., "m" or "ft"). Like other distance measurements, units can be assigned using `units()<-` or `units::set_units`.

Note that a `site` data frame is not required to estimate a distance function. In addition to site IDs and

length, the *sites* data set can contain covariates that are constant for detections on the same transect. We call such covariates site-level covariate to distinguish them from detection-level covariates that vary by detection.

For reference, the first six rows of the sparrow *site* data set are:

```
head(sparrowSiteData)
```

```
##   siteID  length observer bare herb shrub height shrubclass
## 1     A1 500 [m]     obs4 36.7 15.9  20.1   26.4       High
## 2     A2 500 [m]     obs4 38.7 16.1  19.3   25.0       High
## 3     A3 500 [m]     obs5 37.7 18.8  19.8   27.0       High
## 4     A4 500 [m]     obs5 37.7 17.9  19.9   27.1       High
## 5     B1 500 [m]     obs3 58.5 17.6   5.2   19.6        Low
## 6     B2 500 [m]     obs3 56.6 18.1   5.2   19.0        Low
```

We will use `siteID` and `length` in this tutorial.

3: Fit a detection function

After input data are prepared, we recommend plotting a histogram of perpendicular distances to check the minimum, maximum, and units of the distance measurements.

```
hist(sparrowDetectionData$dist
     , col="grey"
     , main=""
     , xlab = "Distance")
rug(sparrowDetectionData$dist,quiet = TRUE)
```

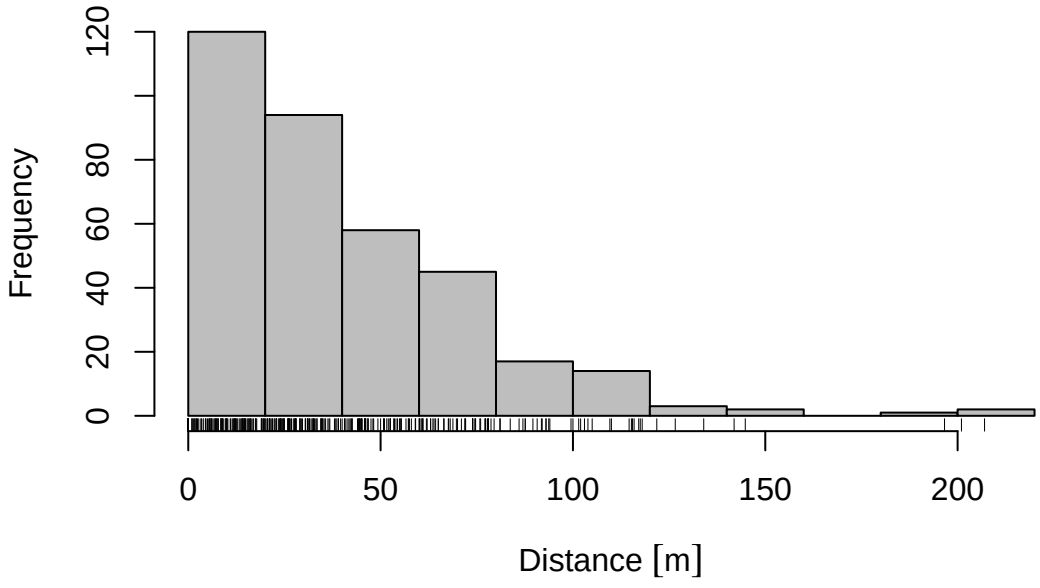

Figure 1: Histogram of sparrow off-transect detection distances.

```
summary(sparrowDetectionData$dist)
```

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

Measurement units appear in the x-axis label of the plot. After a distance function is estimated, which we do next, `Rdistance`'s plotting facilities can improve this histogram and show fit of the distance function to the off-transect distances.

We fit a detection function to perpendicular distances using the function `dfuncEstim`. Function `dfuncEstim` uses maximum likelihood methods to estimate parameters of the distance function. Histogram bin sizes do not matter for parameter estimation and are for visual purposes only. In this tutorial, we proceed using the half-normal distance function. Normally, model selection would take place and a final distance function would be established; but, model selection is outside the scope of this tutorial. Function `autoDistSamp` automates the process of fitting different detection functions, assessing their fit (by AICc), and estimating abundance from the best-fitting function.

Group Sizes

If group sizes are not specified, `Rdistance` assumes all detected groups consisted of one individual. The team collecting our example data occasionally detected two or three sparrows together. Group sizes should be entered in the *detection* data frame, and in our example data are contained in the `groupsize` column. We specify group sizes during distance function estimation as an `groupsize()` term in the formula for the distance function. Group sizes are not used in `dfuncEstim` to estimate parameters of the distance function, but will be used later in function `abundEstim` to estimate density and abundance.

Many distance analysts advocate dropping a small proportion of large distances to improve distance function estimation and reduce variance. Large but rare distances add variance to final estimates and can unduly influence (i.e., bias) distance functions, especially in small (e.g., $n < 100$) data sets. Dropping large observations establishes a maximum and is called right-truncation. Analysts typically drop between one and five percent of the largest distances depending on histogram shape and personal preference. The 95-th quantile of the sparrow detection distances is 104.18, while the 99-th quantile is 143.2. We will right-truncate the sparrow detection distances at 150 meters, which is a nice round number just above the 99-th quantile. In `Rdistance`, we right-truncate by specifying a value for parameter `w.hi`. Parameter `w.hi` must have measurement units because it is a distance.

Distance function estimation

We are ready to estimate a half-normal distance function with right-truncation at 150 m and non-unity group sizes on the sparrow data, i.e.,

```
rightTruncDistance <- units::set_units(150, "m")
dfuncSparrow<- dfuncEstim(formula = dist ~ 1 + offset(groupsize)
                          , detectionData = sparrowDetectionData
                          , likelihood = "halfnorm"
                          , w.hi = rightTruncDistance)
dfuncSparrow
```

```
## Call: dfuncEstim(formula = dist ~ 1 + offset(groupsize), detectionData
##    = sparrowDetectionData, likelihood = "halfnorm", w.hi =
##    rightTruncDistance)
## Coefficients:
##        Estimate  SE        z         p(>|z|)
## Sigma  49.87369  2.014173  24.76138  2.338191e-135
##
## Convergence: Success
```

```
## Function: HALFNORM
## Strip: 0 [m] to 150 [m]
## Effective strip width (ESW): 62.34277 [m]
## Probability of detection: 0.4156185
## Scaling: g(0 [m]) = 1
## Negative log likelihood: 1630.716
## AICc: 3263.443
```

The printout of a distance function object provides parameter estimates, convergence messages, fit information, and effective sampling distance. The estimated parameter of this half-normal distance function is 49.87. As an aside, when a half-normal distance function is estimated, the estimated parameter is the standard deviation of a normal distribution if it were fitted only to positive data and if we required the mean to be exactly zero. Approximately 68 percent of distances will fall between 0 and this parameter. Approximately 95 percent of distances will fall between 0 and twice this parameter. In the sparrow detection data, 69.4 percent of the observed distances are between 0 and 49.87 meters, while 93.5 percent are between 0 and 2(49.87) = 99.75 meters. The closeness of these observed proportions to their theoretical value provides information on fit of the detection function. Further comments on fit are outside the scope of this vignette.

The observation strip comprising the left-, and right-truncation levels appears in the line labeled 'Strip:'. The Effective Strip Width (ESW) is a key piece of information needed to estimate abundance and it appears on the line labeled 'Effective strip width (ESW):'. ESW appears in default `Rdistance` printouts and can be calculated separately using the `ESW()` function. ESW is the distance at which the same number of targets are missed nearer to the observer as were sighted farther from the observer. Missed targets and farther away targets balance themselves out at the ESW. Another way to say think of ESW is that a survey with imperfect detection and ESW equal to X will effectively cover the same area as a study with perfect detection out to a distance of X. See `help(ESW)` for details.

Some researchers prefer to report overall probability of detection in the strip. Probability of detection is ESW divided by the observation strip width. Probability of detection is reported on the line below ESW and in this case equals 62.34 / (150 - 0) = 0.4156. We leave numbers reported in the remainder of the output for later tutorials.

A visual picture of the distance function is obtained using the `plot` method,

```
# Figure 2
plot(dfuncSparrow, nbins =40, col="grey")
```

In Figure 2, the histogram of distances is grey and the estimated distance function is red. ESW is area under the red curve. `Rdistances`'s plot methods are extensive. See **Examples** in `help(plot.dfunc)` examples for ways to modify bins, colors, line types, labels, etc.

4: Estimate abundance from the detection function

Abundance estimation requires additional information contained in the *site* data set. Here, we estimate abundance on the 4105 [km^2] study area using function `abundEstim()`. Confidence intervals for true abundance are calculated using bias-corrected bootstrapping methods (see `help(abundEstim)`). As a result, confidence intervals will vary slightly between runs due to simulation error.

```
fit <- abundEstim(dfuncSparrow
                , detectionData = sparrowDetectionData
                , siteData = sparrowSiteData
                , area = saSize
                , ci = 0.95
                )
```

```
fit
```

```
## Call: dfuncEstim(formula = dist ~ 1 + offset(groupsize), detectionData
```

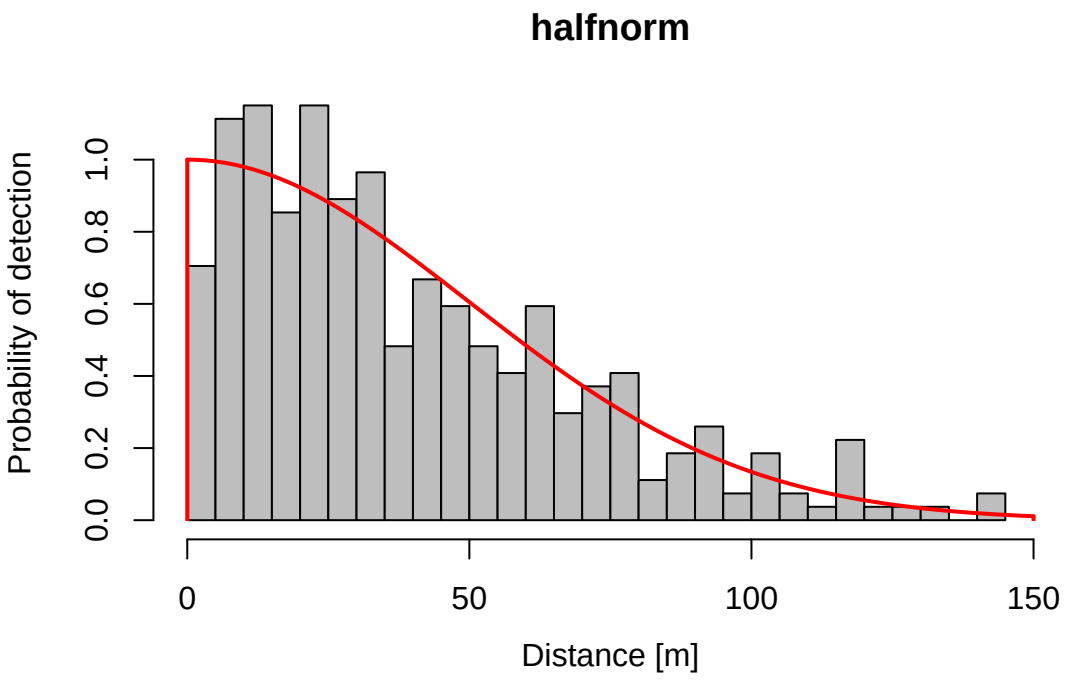

Figure 2: The half-normal distance function fitted to sparrow off-transect detection distances.

```
##    = sparrowDetectionData, likelihood = "halfnorm", w.hi =
##    rightTruncDistance)
## Coefficients:
##        Estimate  SE        z         p(>|z|)
## Sigma  49.87369  2.014173  24.76138  2.338191e-135
##
## Convergence: Success
## Function: HALFNORM
## Strip: 0 [m] to 150 [m]
## Effective strip width (ESW): 62.34277 [m]
##                       95% CI: 56.66586 [m] to 68.72937 [m]
## Probability of detection: 0.4156185
## Scaling: g(0 [m]) = 1
## Negative log likelihood: 1630.716
## AICc: 3263.443
##
##      Surveyed Units: 36000 [m]
##    Individuals seen: 371 in 353 groups
##  Average group size: 1.050992
##               Range: 1 to 3
##
## Density in sampled area: 8.265237e-05 [1/m^2]
##                  95% CI: 6.544368e-05 [1/m^2] to 0.0001003868 [1/m^2]
##
## Abundance in 4.105e+09 [m^2] study area: 339288
##                                  95% CI: 268646.3 to 412088
```

The estimated number of sparrows on the 4105 [km^2] study area is 339,288 with 95 percent confidence interval from 268,646 to 412,088 individuals. Estimated density is 8.265237e-05 [1/m^2], which converts to 0.8265 [1/ha] (95 percent CI: 0.6544 [1/ha] to 1.0039 [1/ha]). The observer's effective strip width was 62.3 [m] (95 percent CI: 56.67 [m] to 68.73 [m]).

5: Advanced Remarks

Bootstrap Distribution

`Rdistance` stores bootstrap parameter values, density, and ESW values in the `$B` component of abundance objects. Storing bootstrap iterations provides flexibility for more advanced analyses. For example,

- Users can combine bootstrap iterations from multiple abundance objects by appending `$B` components together. New confidence intervals from the new `$B` component will be computed when the abundance object is printed.
- Users can compute confidence intervals using different methods (e.g., percentile instead of bia-corrected).
- Users can compute the variance of other quantities that depend on parameters, density, or ESW; for example, density in a particular strip.

- Users can plot the full bootstrap distribution of density (or ESW) to inspect statistical properties.

The first six lines of the bootstrap iterations are:

```
# Convert to hectares for readability
units(fit$B$density) <- "1/ha"

head(fit$B)
```

```
##            density  effDistance
## 1 0.7411862 [1/ha] 65.96033 [m]
## 2 0.8697759 [1/ha] 64.51214 [m]
## 3 0.8209953 [1/ha] 62.59340 [m]
## 4 0.8248241 [1/ha] 60.78737 [m]
## 5 0.9833776 [1/ha] 57.05958 [m]
## 6 0.9260318 [1/ha] 58.79328 [m]
```

The bootstrap distribution of sparrow density is,

```
# Figure 3
hist(fit$B$density
     , n = 30
     , xlab = "Density"
     , main = NULL)
# Show final density estimates, after converting to 1/ha
d <- fit$density
d.ci <- fit$density.ci
units(d) <- "1/ha"
units(d.ci) <- "1/ha"
abline(v = c(d, d.ci), col="blue")
```

In Figure 3, the density point estimate (0.8265237 [1/ha]) and confidence limits (0.6544368 [1/ha] to 1.0038680 [1/ha]) visually plot where we expect and encompass the center of the bootstrap distribution. The same figure for other data may show final estimates 'shifted' relative to the bootstrap distribution because the bias-corrected CI method attempts to remove estimation bias (i.e., removes distributional 'shift').

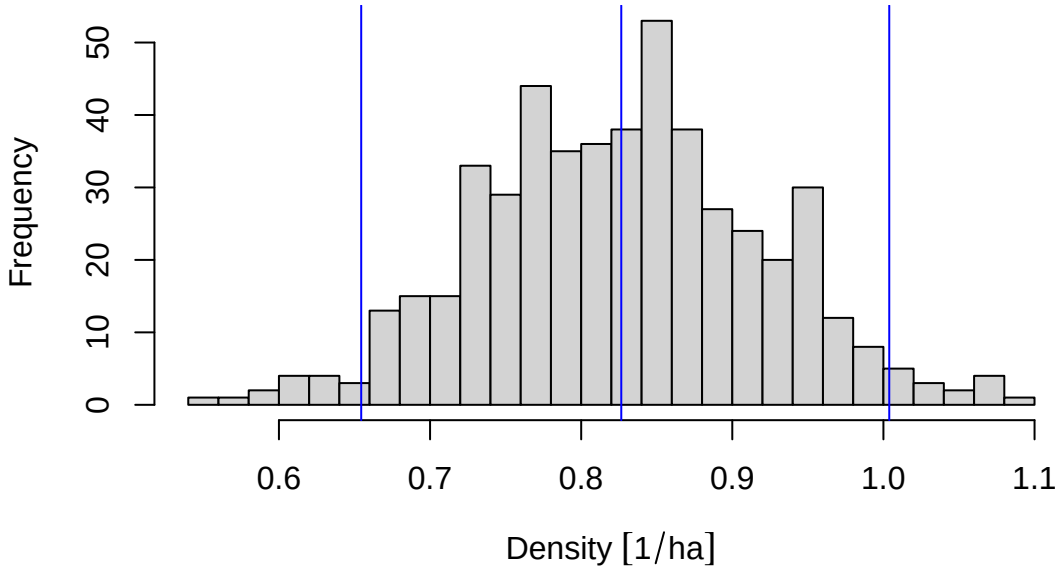

Figure 3: Bootstrap distribution of sparrow density computed in the `abundEstim` routine.

Output Measurment Units

It is possible to change the output measurement units on ESW and density by setting the `outputUnits =` parameter in the original call to `dfuncEstim`. Setting `outputUnits` only effects reporting units (output). There is no need to change the units of inputs because conversion is handled internally and automatically. For example, to output linear distances in kilometers, and density in square kilometers, issue the following:

```
dfuncSparrow <- dfuncEstim(formula = dist~1
                           , detectionData = sparrowDetectionData
                           , likelihood = "halfnorm"
                           , w.hi = rightTruncDistance
                           , outputUnits = "km")
```

```
fit <- abundEstim(dfuncSparrow
                  , detectionData = sparrowDetectionData
                  , siteData = sparrowSiteData
                  , area = saSize
                  , ci = 0.95
                  )
```

```
fit
```

```
## Call: dfuncEstim(formula = dist ~ 1, detectionData =
##    sparrowDetectionData, likelihood = "halfnorm", w.hi =
##    rightTruncDistance, outputUnits = "km")
## Coefficients:
##        Estimate    SE           z         p(>|z|)
## Sigma  0.04987416  0.002014229  24.76092  2.365001e-135
```

```
##
## Convergence: Success
## Function: HALFNORM
## Strip: 0 [km] to 0.15 [km]
## Effective strip width (ESW): 0.06234334 [km]
##                       95% CI: 0.05674188 [km] to 0.06864928 [km]
## Probability of detection: 0.4156223
## Scaling: g(0 [km]) = 1
## Negative log likelihood: -807.7218
## AICc: -1613.432
##
##      Surveyed Units: 36 [km]
##    Individuals seen: 353 in 353 groups
##  Average group size: 1
##               Range: 1 to 1
##
## Density in sampled area: 78.64156 [1/km^2]
##                  95% CI: 60.93035 [1/km^2] to 95.15893 [1/km^2]
##
## Abundance in 4105 [km^2] study area: 322823.6
##                              95% CI: 250119.1 to 390627.4
```

Additional Reading

View the full set of vignettes, as well as other teaching and tutorial materials, on the `Rdistance` wiki: https://github.com/tmcd82070/Rdistance/wiki. Suggest ideas for vignettes and wiki pages, or submit helpful examples, by creating an Issue on the `Rdistance` GitHub site.