Migrating distance sampling projects from Distance for Windows to the Distance R package

Eric Rexstad, David L. Miller, Laura Marshall and Len Thomas

Centre for Research into Ecological and Environmental Modelling University of St Andrews

Introduction

The Distance software (Thomas et al., 2010) has been downloaded >40,000 times in its 20-year history. Much of the underlying machinery is written in R. For some users, there may be benefits to performing the analysis with the underlying R code, rather than working with the graphical user interface (GUI).

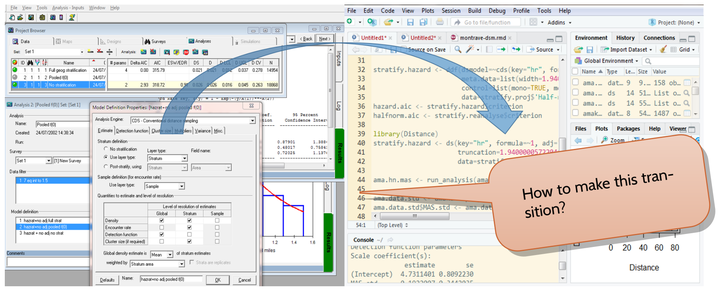

Fig. 1: Traditional Distance for Windows interface (left) and distance sampling analysis in R (right).

Challenges hindering the transition between analysis with the GUI and analyses in R are two-fold:

- Legacy data reside in Distance (GUI) projects, unavailable for importing into R, and
- Analyses that are easily described using the GUI may be difficult to specify, particularly if analyst is not proficient in R.

How to bridge between the two?

Distance GUI projects contain essential information necessary to conduct an analysis. The fundamental purpose of the `readdst` package is to access this information and place it into R objects for further scrutiny.

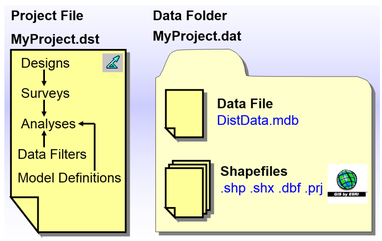

Figure 1: Database structure of a Distance project.

The elements of a distance sampling analysis are contained inside Distance project files in the form of an Access database.

1. Data
2. model definitions
3. analysis results

Mining information from tables within the database, via either `RODBC` (Windows) or `mdb-tools` (Macs). Then it is a matter of performing the following three steps.

- Trick is to extract contents of (2) and translate into R code
- Target the results of previous step to contents of (1) the data
- Perhaps contrast results of R analysis with results stored in the Access database

Applications–Legacy projects

Workhorse function in `readdst` package is `convert_projects()`. Its single argument is simply the complete location path to an existing Distance GUI project (created by either Distance 6 or 7). `convert_projects()` interrogates the Access database and translates the contents of data base tables into an R object of class `converted_distance_analyses`, a list of lists of class `converted_distance_analysis`. Fig. 3 gives an outline of the content of this list.

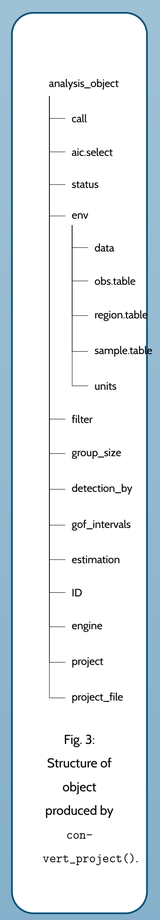

Fig. 3: Structure of object produced by `convert_project()`.

Object produced by `convert_project()`

Figure 3 (left) shows the structure of an object of type `converted_distance_analysis`. There are as many of these produced by `convert_project()` as there were analyses in the Distance project.
The list contains all the salient information of a Distance analysis extracted from the Access database. We point out several of the list elements critical to subsequent processing of the now-extracted data.

call R code call to the function `mrds()` to duplicate the analyses done in the Distance GUI.

env An *environment* consisting of a set of data frames. These data frames include the original data extracted from the Distance project, along with the observation, sample and region tables describing the hierarchial nature of the data base.

The nature of the analysis conducted by the Distance GUI, translated into a call to `mrds()` (Laake et al., 2018), as well as the data used in the analysis is contained within this object and available for further analysis within the R environment.
The `converted_distance_analysis` list can be passed as an argument to `run_analysis()`. This function will perform an `mrds()` analysis *behind-the-scenes* , without the user seeing how it was performed.

Learning structure of R interface

A user familiar with performing distance sampling analyses in the Distance GUI can use `converted_distance_analysis` objects to learn how to perform corresponding analyses using `mrds()`. Alternatively, the imported data can be analysed using the Distance (Miller, 2018) package. Examples of both approaches provided below

```
library(readdst)
home.dir <- path.expand("~")
stratify.proj.directory <- system.file("Stratify", package="readdst")
stratify.proj.name <- paste0(stratify.proj.directory, "/Vignette-stratify")
stratify.proj <- convert_project(stratify.proj.name)
# post-conversion, re-run analysis with 'run_analysis'
strat.mrds.a1 <- run_analysis(stratify.proj$`Half-normal_cosine_no_stratification_exact`)
# perform same analysis manually, using ds()
library(Distance)
strat.dist.a1 <- ds(key="hn", formula=~1, adj="cos", order=2,
                    data=stratify.proj$`Half-normal_cosine_no_stratification_exact`$env$data)
```

Comparative analysis of difficult data

- We try to do this. what?
- we try to do that.

Caveats

`readdst` is not able to translate all GUI analyses into R code. Current limitations are inability to translate

- analyses using the `dsm`, `mads` and `Dssim` engines,
- analyses using post-stratification and
- bootstraps for variance estimation.

Additional information

References

Laake, J., D. Borchers, D. Miller, and J. Bishop. 2018. *package mrds*.

Miller, D.. 2018. *package Distance*.

Miller, D. L.. 2017. *Package readdst*.

Miller, D. L., E. Rexstad, L. Thomas, L. Marshall, and J. Laake. 2016. Distance Sampling in R. *bioRxiv*.

Thomas, L., S. T. Buckland, E. A. Rexstad, J. L. Laake, S. Strindberg, S. L. Hedley, J. R. Bishop, T. A. Marques, and K. P. Burnham. 2010. Distance software: design and analysis of distance sampling surveys for estimating population size. *Journal of Applied Ecology*, 47(1):5–14.

QR codes to package/website/bioRxiv