Package 'Distance'

September 16, 2014

Maintainer David Lawrence Miller <dave@ninepointeightone.net>

License GPL (>= 2)

Title A simple way to fit detection functions to distance sampling data and calculate abundance/density for biological populations.

LazyLoad yes

Author David Lawrence Miller

Description A library that provides a simple way of fitting detection functions to distance sampling data for both line and point transects. Adjustment term selection, left and right truncation as well as monotonicity constraints and binning are supported. Abundance and density estimates can also be calculated if survey area information is provided.

Version 0.9.2

URL http://github.com/DistanceDevelopment/Distance/

Date 2014-10-15

Depends R (>= 3.0), mrds (>= 2.1.8)

Suggests testthat

NeedsCompilation no

Repository CRAN

Date/Publication 2014-09-16 02:14:01

R topics documented:

`Distance-package` *Distance sampling*

Description

`Distance` is a simple way to fit detection functions and estimate abundance using distance sampling methodology.

Details

Underlying `Distance` is the package `mrds`, for more advanced analyses (such as those involving double observer surveys) one may find it necessary to use `mrds`.

Author(s)

David L. Miller <dave@ninepointeightone.net>

References

Laake, J.L. and D.L. Borchers. 2004. Methods for incomplete detection at distance zero. In: Advanced Distance Sampling, eds. S.T. Buckland, D.R.Anderson, K.P. Burnham, J.L. Laake, D.L. Borchers, and L. Thomas. Oxford University Press.

Marques, F.F.C. and S.T. Buckland. 2004. Covariate models for the detection function. In: Advanced Distance Sampling, eds. S.T. Buckland, D.R.Anderson, K.P. Burnham, J.L. Laake, D.L. Borchers, and L. Thomas. Oxford University Press.

`checkdata` *Check that the data supplied to* `ds` *is correct*

Description

This is an internal function that checks the `data.frames` supplied to `ds` are "correct".

Usage

```
checkdata(data, region.table = NULL, sample.table = NULL,
  obs.table = NULL)
```

Arguments

data	as in ds
sample.table	as in ds
region.table	as in ds
obs.table	as in ds

Value

Throws an error if something goes wrong, otherwise returns a data.frame.

Author(s)

David L. Miller

---

create.bins *Create bins from a set of binned distances and a set of cutpoints.*

Description

This is an internal routine and shouldn't be necessary in normal analyses.

Usage

```
create.bins(data, cutpoints)
```

Arguments

data	data.frame with at least the column distance.
cutpoints	vector of cutpoints for the bins

Value

data data with two extra columns distbegin and distend.

Author(s)

David L. Miller

ds *Fit detection functions and calculate abundance from line or point transect data*

Description

This function fits detection functions to line or point transect data and then (provided that survey information is supplied) calculates abundance and density estimates. The examples below illustrate some basic types of analysis using `ds()`.

Usage

```
ds(data, truncation = ifelse(is.null(cutpoints), ifelse(is.null(data$distend),
  max(data$distance), max(data$distend)), max(cutpoints)),
  transect = c("line", "point"), formula = ~1, key = c("hn", "hr",
  "unif"), adjustment = c("cos", "herm", "poly"), order = NULL,
  scale = c("width", "scale"), cutpoints = NULL, dht.group = FALSE,
  monotonicity = ifelse(formula == ~1, "strict", "none"),
  region.table = NULL, sample.table = NULL, obs.table = NULL,
  convert.units = 1, method = "nlminb", quiet = FALSE, debug.level = 0,
  initial.values = NULL)
```

Arguments

`data`	a `data.frame` containing at least a column called `distance`. NOTE! If there is a column called `size` in the data then it will be interpreted as group/cluster size, see the section "Clusters/groups", below. One can supply data as a "flat file" and not supply `region.table`, `sample.table` and `obs.table`, see "Data format", below and `flatfile`.
`truncation`	either truncation distance (numeric, e.g. 5) or percentage (as a string, e.g. "15%"). Can be supplied as a `list` with elements `left` and `right` if left truncation is required (e.g. `list(left=1,right=20)` or `list(left="1%",right="15%")` or even `list(left="1",right="15%")`). By default for exact distances the maximum observed distance is used as the right truncation. When the data is binned, the right truncation is the largest bin end point. Default left truncation is set to zero.
`transect`	indicates transect type "line" (default) or "point".
`formula`	formula for the scale parameter. For a CDS analysis leave this as its default `~1`.
`key`	key function to use; "hn" gives half-normal (default), "hr" gives hazard-rate and "unif" gives uniform. Note that if uniform key is used, covariates cannot be included in the model.
`adjustment`	adjustment terms to use; "cos" gives cosine (default), "herm" gives Hermite polynomial and "poly" gives simple polynomial. "cos" is recommended. A value of `NULL` indicates that no adjustments are to be fitted.

`order` orders of the adjustment terms to fit (as a vector/scalar), the default value (`NULL`) will select via AIC. For cosine adjustments, valid orders are integers greater than 2 (except when a uniform key is used, when the minimum order is 1). For Hermite polynomials, even integers equal or greater than 4 are allowed. For simple polynomials even integers equal or greater than 2 are allowed.

`scale` the scale by which the distances in the adjustment terms are divided. Defaults to "width", scaling by the truncation distance. If the key is uniform only "width" will be used. The other option is "scale": the scale parameter of the detection

`cutpoints` if the data are binned, this vector gives the cutpoints of the bins. Ensure that the first element is 0 (or the left truncation distance) and the last is the distance to the end of the furthest bin. (Default `NULL`, no binning.) Note that if `data` has columns `distbegin` and `distend` then these will be used as bins if `cutpoints` is not specified. If both are specified, `cutpoints` has precedence.

`monotonicity` should the detection function be constrained for monotonicity weakly (`"weak"`), strictly (`"strict"`) or not at all (`"none"` or `FALSE`). See Montonicity, below. (Default `"strict"`). By default it is on for models without covariates in the detection function, off when covariates are present.

`dht.group` should density abundance estimates consider all groups to be size 1 (abundance of groups) `dht.group=TRUE` or should the abundance of individuals (group size is taken into account), `dht.group=FALSE`. Default is `FALSE` (abundance of individuals is calculated).

`region.table` `data.frame` with two columns:

`Region.Label`	label for the region
`Area`	area of the region

`region.table` has one row for each stratum. If there is no stratification then `region.table` has one entry with `Area` corresponding to the total survey area.

`sample.table` `data.frame` mapping the regions to the samples (i.e. transects). There are three columns:

`Sample.Label`	label for the sample
`Region.Label`	label for the region that the sample belongs to.
`Effort`	the effort expended in that sample (e.g. transect length).

`obs.table` `data.frame` mapping the individual observations (objects) to regions and samples. There should be three columns:

`object`	
`Region.Label`	label for the region that the sample belongs to.
`Sample.Label`	label for the sample

`convert.units` conversion between units for abundance estimation, see "Units", below. (De-

	faults to 1, implying all of the units are "correct" already.)
method	optimization method to use (any method usable by optim or **optimx**). Defaults to "nlminb".
debug.level	print debugging output. 0=none, 1-3 increasing level of debugging output.
quiet	surpress non-essential messages (useful for bootstraps etc). Default value FALSE.
initial.values	a list of named starting values, see mrds-opt. Only allowed when AIC term selection is not used.

Value

a list with elements:

ddf	a detection function model object.
dht	abundance/density information (if survey region data was supplied, else NULL).

Details

If abundance estimates are required then the data.frames region.table and sample.table must be supplied. If data does not contain the columns Region.Label and Sample.Label thenthe data.frame obs.table must also be supplied. Note that stratification only applies to abundance estimates and not at the detection function level.

Clusters/groups

Note that if the data contains a column named size and region.table, sample.table and obs.table are supplied, cluster size will be estimated and density/abundance will be based on a clustered analsis of the data. Setting this column to be NULL will perform a non-clustred analysis (for example if "size" means something else in your dataset).

Truncation

The right truncation point is by default set to be largest observed distance or bin end point. This is a default will not be appropriate for all data and can often be the cause of model convergence failures. It is recommended that one plots a histogram of the observed distances prior to model fitting so as to get a feel for an appropriate truncation distance. (Similar arguments go for left truncation, if appropriate). Buckland et al (2001) provide guidelines on truncation.

When specified as a percentage, the largest right and smallest left percent distances are discarded. Percentages cannot be supplied when using binned data.

Binning

Note that binning is performed such that bin 1 is all distances greater or equal to cutpoint 1 (>=0 or left truncation distance) and less than cutpoint 2. Bin 2 is then distances greater or equal to cutpoint 2 and less than cutpoint 3 and so on.

Monotonicity

When adjustment terms are used, it is possible for the detection function to not always decrease with increasing distance. This is unrealistic and can lead to bias. To avoid this, the detection function can be constrained for monotonicity (and is by default for detection functions without covariates).

Monotonicity constraints are supported in a similar way to that described in Buckland et al (2001). 20 equally spaced points over the range of the detection function (left to right truncation) are evaluated at each round of the optimisation and the function is constrained to be either always less than it's value at zero (`"weak"`) or such that each value is less than or equal to the previous point (monotonically decreasing; `"strict"`). See also `check.mono` in `mrds`.

Even with no monotonicity constraints, checks are still made that the detection function is monotonic, see `check.mono`.

Units

In extrapolating to the entire survey region it is important that the unit measurements be consistent or converted for consistency. A conversion factor can be specified with the `convert.units` variable. The values of `Area` in `region.table`, must be made consistent with the units for `Effort` in `sample.table` and the units of `distance` in the `data.frame` that was analyzed. It is easiest if the units of `Area` are the square of the units of `Effort` and then it is only necessary to convert the units of `distance` to the units of `Effort`. For example, if `Effort` was entered in kilometers and `Area` in square kilometers and `distance` in meters then using `convert.units=0.001` would convert meters to kilometers, density would be expressed in square kilometers which would then be consistent with units for `Area`. However, they can all be in different units as long as the appropriate composite value for `convert.units` is chosen. Abundance for a survey region can be expressed as: `A*N/a` where `A` is `Area` for the survey region, `N` is the abundance in the covered (sampled) region, and `a` is the area of the sampled region and is in units of `Effort * distance`. The sampled region `a` is multiplied by `convert.units`, so it should be chosen such that the result is in the same units as `Area`. For example, if `Effort` was entered in kilometers, `Area` in hectares (100m x 100m) and `distance` in meters, then using `convert.units=10` will convert `a` to units of hectares (100 to convert meters to 100 meters for distance and .1 to convert km to 100m units).

Data format

One can supply `data` only to simply fit a detection function. However, if abundance/density estimates are necessary further information is required. Either the `region.table`, `sample.table` and `obs.table` `data.frames` can be supplied or all data can be supplied as a "flat file" in the `data` argument. In this format each row in data has additional information that would ordinarily be in the other tables. This usually means that there are additional columns named: `Sample.Label`, `Region.Label`, `Effort` and `Area` for each observation. See `flatfile` for an example.

Author(s)

David L. Miller

References

Buckland, S.T., Anderson, D.R., Burnham, K.P., Laake, J.L., Borchers, D.L., and Thomas, L. (2001). Distance Sampling. Oxford University Press. Oxford, UK.

Buckland, S.T., Anderson, D.R., Burnham, K.P., Laake, J.L., Borchers, D.L., and Thomas, L. (2004). Advanced Distance Sampling. Oxford University Press. Oxford, UK.

See Also

flatfile

Examples

```
# An example from mrds, the golf tee data.
library(Distance)
data(book.tee.data)
tee.data<-book.tee.data$book.tee.dataframe[book.tee.data$book.tee.dataframe$observer==1,]
ds.model<-ds(tee.data,4)
summary(ds.model)
plot(ds.model)

## Not run:
# same model, but calculating abundance
# need to supply the region, sample and observation tables
region<-book.tee.data$book.tee.region
samples<-book.tee.data$book.tee.samples
obs<-book.tee.data$book.tee.obs

ds.dht.model<-ds(tee.data,4,region.table=region,
             sample.table=samples,obs.table=obs)
summary(ds.dht.model)

# specify order 2 cosine adjustments
ds.model.cos2<-ds(tee.data,4,adjustment="cos",order=2)
summary(ds.model.cos2)

# specify order 2 and 3 cosine adjustments, turning monotonicity
# constraints off
ds.model.cos23<-ds(tee.data,4,adjustment="cos",order=c(2,3),
                   monotonicity=FALSE)
# check for non-monotonicity -- actually no problems
check.mono(ds.model.cos23$ddf,plot=TRUE,n.pts=100)

# include both a covariate and adjustment terms in the model
ds.model.cos2.sex <- ds(tee.data,4,adjustment="cos",order=2,
                        monotonicity=FALSE, formula=~as.factor(sex))
# check for non-monotonicity -- actually no problems
check.mono(ds.model.cos2.sex$ddf,plot=TRUE,n.pts=100)

# truncate the largest 10% of the data and fit only a hazard-rate
# detection function
ds.model.hr.trunc<-ds(tee.data,truncation="10%",key="hr",adjustment=NULL)
summary(ds.model.hr.trunc)

## End(Not run)
```

flatfile *Flat files*

Description

Distance allows loading data as a "flat file" and analyse data (and obtain abundance estimates) straight away, provided that the format of the flat file is correct. One can provide the file as, for example, an Excel spreadsheet using read.xls in **gdata** or CSV using read.csv.

Details

Each row of the data table corresponds to one observation and must have a the following columns:

distance	observed distance to object
Sample.Label	Identifier for the sample (transect id)
Effort	effort for this transect (e.g. line transect length or number of times point transect was visited)
Region.Label	code for regional strata (see below)
Area	area of the strata

Note that in the simplest case (one area surveyed only once) there is only one Region.Label and a single corresponding Area duplicated for each observation.

The example given below was provided by Eric Rexstad.

Examples

```
library(Distance)
# Need to have the gdata library installed from CRAN, requires a system
# with perl installed (usually fine for Linux/Mac)
library(gdata)

# Need to get the file path first
# Going to the path given in the below, one can examine the format
minke.filepath <- system.file("minke.xlsx",package="Distance")

# Load the Excel file, note that header=FALSE and we add column names after
minke <- read.xls(minke.filepath, stringsAsFactor=FALSE,header=FALSE)
names(minke) <- c("Region.Label", "Area", "Sample.Label", "Effort","distance")
# One may want to call edit(minke) or head(minke) at this point
# to examine the data format

# Due to the way the file was saved and the default behaviour in R
# for numbers stored with many decimal places (they are read as strings
# rather than numbers, see str(minke)). We must coerce the Effort column
# to numeric
minke$Effort <- as.numeric(minke$Effort)

## perform an analysis using the exact distances
```

```
pooled.exact <- ds(minke, truncation=1.5, key="hr", order=0)
summary(pooled.exact)


## Try a binned analysis
# first define the bins
dist.bins <- c(0,.214, .428,.643,.857,1.071,1.286,1.5)
pooled.binned <- ds(minke, truncation=1.5, cutpoints=dist.bins, key="hr", order=0)

# binned with stratum as a covariate
minke$stratum <- ifelse(minke$Region.Label=="North", "N", "S")
strat.covar.binned <- ds(minke, truncation=1.5, key="hr",
                         formula=~as.factor(stratum), cutpoints=dist.bins)

# Stratified by North/South
full.strat.binned.North <- ds(minke[minke$Region.Label=="North",],
                  truncation=1.5, key="hr", order=0, cutpoints=dist.bins)
full.strat.binned.South <- ds(minke[minke$Region.Label=="South",],
                     truncation=1.5, key="hr", order=0, cutpoints=dist.bins)

## model summaries
model.sel.bin <- data.frame(name=c("Pooled f(0)", "Stratum covariate",
                                   "Full stratification"),
                            aic=c(pooled.binned$ddf$criterion,
                                  strat.covar.binned$ddf$criterion,
                                  full.strat.binned.North$ddf$criterion+
                                  full.strat.binned.South$ddf$criterion))

# Note model with stratum as covariate is most parsimonious
print(model.sel.bin)
```

`minke` *Simulated minke whale data*

Description

Data simulated from models fitted to 1992/1993 Southern Hemisphere minke whale data collected by the International Whaling Commission. See Branch and Butterworth (2001) for survey details (survey design is shown in figure 1(e)). Data simulated by David Borchers.

Format

`data.frame` with 99 observations of 5 variables:

`Region.Label`	stratum label (`"North"` or `"South"`)
`Area`	stratum area
`Sample.Label`	transect identifier
`Effort`	transect length
`distance`	observed distance

Details

Data are included here as both R data and as an Excel spreadsheet to illustrate the "flat file" input method. See `flatfile` for how to load this data and an example analysis.

Source

Shipped with the DISTANCE Windows application.

References

Branch, T.A. and D.S. Butterworth (2001) Southern Hemisphere minke whales: standardised abundance estimates from the 1978/79 to 1997/98 IDCR-SOWER surveys. Journal of Cetacean Research and Management 3(2): 143-174

Hedley, S.L., and S.T. Buckland. Spatial Models for Line Transect Sampling. Journal of Agricultural, Biological, and Environmental Statistics 9, no. 2 (2004): 181-199. doi:10.1198/1085711043578.

Examples

```
data(minke)
head(minke)
```

---

plot.dsmodel *Plot a fitted detection function*

Description

This is just a simple wrapper around `plot.ds`. See the manual page for that function for more information.

Usage

```
## S3 method for class 'dsmodel'
plot(x, ...)
```

Arguments

`x`	an object of class `dsmodel`.
`...`	extra arguments to be passed to `plot.ds`.

Value

`NULL`, just produces a plot.

Author(s)

David L. Miller

print.dsmodel *Simple pretty printer for distance sampling analyses*

Description

Simply prints out a brief description of the model which was fitted. For more detailed information use summary.

Usage

```
## S3 method for class 'dsmodel'
print(x, ...)
```

Arguments

x	a distance sampling analysis (result from calling ds).
...	not passed through, just for S3 compatibility.

Author(s)

David L. Miller

print.summary.dsmodel *Print summary of distance detection function model object*

Description

Provides a brief summary of a distance sampling analysis. Including: detection function parameters, model selection criterion, and optionally abundance in the covered (sampled) region and its standard error.

Usage

```
## S3 method for class 'summary.dsmodel'
print(x, ...)
```

Arguments

x	a summary of distance sampling analysis
...	unspecified and unused arguments for S3 consistency

Value

Nothing, just prints the summary.

Author(s)

David L. Miller and Jeff Laake

See Also

summary.ds

---

summary.dsmodel *Summary of distance sampling analysis*

Description

Provides a brief summary of a distance sampling analysis. This includes

Usage

```
## S3 method for class 'dsmodel'
summary(object, ...)
```

Arguments

object	a distance analysis
...	unspecified and unused arguments for S3 consistency

Value

list of extracted and summarized objects

Note

This function just calls summary.ds and dht, collates and prints the results in a nice way.

Author(s)

David L. Miller

Index