Package 'mmds'

April 11, 2012

Version 1.1

Date 2011-07-10

Title Mixture Model Distance Sampling (mmds)

Author David Lawrence Miller <dave@ninepointeightone.net>

Maintainer David Lawrence Miller <dave@ninepointeightone.net>

Description This library implements mixture model distance sampling methods. See Miller and Thomas (in prep.).

Depends R (>= 2.8.0)

LazyLoad yes

License GPL-2

Collate 'checkinitialvalues.R' 'DeltaMethod.R' 'detfct.R' 'em.R''eval.pdf.R' 'fitmix.R' 'flt.gr.R' 'flt.R' 'flt.var.R''getpars.R' 'goft
package.R'

Repository CRAN

Date/Publication 2012-04-11 06:03:52

R topics documented:

`ds.mixture` *A fitted Mixture Model Detection Function Object*

Description

The fitted mixture model detection function object returned by `fitmix`. Knowledge of most of this is not useful. Use `link{summary.ds.mixture}` for result summaries.

Details

A `ds.mixture` object has the following elements:

distance	Vector of distances used in the analysis.
likelihood	Value of the log-likelihood at the maxima.
pars	Parmeter estimates. See `mmds.pars` for more information.
mix.terms	Number of mixture terms fit.
width	Truncation distance used.
z	List containing the matrix of covariates used. Output from `model.matrix`.
zdim	Number of columns of `z`. See `mmds.pars` for more information.
hessian	Hessian matrix at the maxima.
pt	Logical indicating whether the data were from a point transect survey.
data	Data frame after truncation.
ftype	Type of detection function.
ctrl.options	Options passed to `optim`.
showit	Debug level.
opt.method	Optimisation method used.
usegrad	Were analytic gradients used?
model.formula	Model formula.
mu	Per-observation effective trip width/effective area of detection.
pa.vec	Vector of per-observation detectabilities.
N	Estimate of N in the covered area (Horvitz-Thompson).
pa	Average detectability.
pars.se	Standard errors of the parameters.
N.se	Standard error of the Horvitz-Thompson estimate of the abundance.
pa.se	Standard error of the average detectability.
aic	AIC of the fitted model.
cvm	Cramer-von Mises GoF test results. List containing: `p`, the p-value and `W`, the test statistic.
ks	Kolmogorov-Smirnov test results. List containing: `p`, the p-value and `Dn`, the test statistic. See `mmds.gof` for

Note

`ds.mixture` objects can be passed to `step.ds.mixture` to select number of mixture components based on AIC score.

Author(s)

David L. Miller

`fitmix` *Mixture Model Distance Detection Function Fitting*

Description

Fits a mixture of half-normals as a detection function to distance sampling data collected via either line or point transects, possibly with covariates.

Usage

```
fitmix(data, width, mix.terms = 1, pt = FALSE,
  model.formula = "~1", initialvalues = NULL, showit = 0,
  ctrl.options = c(maxit = 10000),
  opt.method = "BFGS+SANN", usegrad = TRUE, ftype = "hn")
```

Arguments

`data`	`data.frame` containing the distances and covariates to be used in the analusis (see Details).
`width`	Truncation distance.
`mix.terms`	Number of mixture components to use. Defaults to 1 (ie. CDS).
`pt`	Is the data from point transects? Default FALSE.
`model.formula`	Formula to be used for the covariates. Defaults to "~1" (ie. no covariates).
`initialvalues`	User supplied initialvalues if needed. Defaults to `NULL`. See `mmds.pars` for more information.
`showit`	Debugging level from 0 to 3, with 3 being most verbose. Defaults to 0.
`ctrl.options`	Options to give to the `optim`. Defaults to `c(maxit=10000)`.
`opt.method`	Optimisation method to use, one of "BFGS", "BFGS+SANN" or "EM". Defaults to "BFGS+SANN", see Details.
`usegrad`	Should analytic derivatives be used in the optimisation? Default TRUE.
`ftype`	Function type to be used as the detection function, currently only "hn".

Value

a `ds.mixture` model object.

Details

This is the main routine that fits mixture model detection functions.

`data` should be a `data.frame` with (at least) a column named `distance`. Any covariates given in `model.formula` should be named in `data`. Note that rows with `distance` greater than `width` will be discarded.

See `step.ds.mixture` for AIC selection for the number of mixture components.

Author(s)

David L. Miller

References

Miller, D.L. and L. Thomas (in prep.). Mixture model distance sampling detection functions.

See Also

summary.ds.mixture step.ds.mixture plot.ds.mixture sim.mix mmds.gof ds.mixture mmds.gof fitmix mmds.pars step.ds.mixture

Examples

```
library(mmds)
set.seed(0)
## simulate some line transect data from a 2 point mixture
sim.dat<-sim.mix(c(-0.223,-1.897,inv.reparam.pi(0.3)),2,100,1)
## fit the model
fit.sim.dat<-fitmix(sim.dat,1,2)
## what happened?
summary(fit.sim.dat)
```

---

getpars *Grab parameter values*

Description

Extract parameter values and create a named list.

Usage

```
getpars(fpar, mix.terms, zdim = 0, z = NULL)
```

Arguments

fpar	The $par element ds.mixture object.
mix.terms	Number of mixture components.
zdim	Number of covariates.
z	Covariate matrix.

Value

a named list with elements $key.scale (giving the key scales) and $mix.prop giving the mixture proportions.

Author(s)

David L. Miller

`inv.reparam.pi` *Reparametrize mixture proportions*

Description

Re-parameterise the mixture proportions so that when there is more than a 2-point mixture, the proportions sum to 1.

Usage

```
inv.reparam.pi(mix.prop, lastpar = FALSE)
```

Arguments

`mix.prop`	A set of mixture proportions.
`lastpar`	Is the last parameter provided, i.e. does sum(mix.prop)=1?

Value

a vector of parameters

Notes

See Miller and Thomas for information on exactly how these are calculated. Thanks go to David Borchers for proposing the trick.

Author(s)

David L. Miller

References

Miller, D.L. and L. Thomas (in prep.). Mixture model distance sampling detection functions.

See Also

reparam.pi

Examples

```
library(mmds)
reparam.pi(inv.reparam.pi(0.3))
reparam.pi(inv.reparam.pi(c(0.3,0.4,0.1),lastpar=TRUE))
```

mmds.gof *Goodness of fit for mixture model detection functions*

Description

Goodness of fit testing for detection for mixture model detection functions.

Details

Two goodness of fit tests are provided: the Cramer-von Mises and the Kolmogorov-Smirnov. Both are implemented as in Buckland et al. (2004).

Print methods are provided, so accessing the `ks` and `cvm` elements of a `ds.mixture` object will give suitable summaries.

David L. Miller

mmds.pars *Parameters in* mmds

Description

The internal parametrisation used in `mmds` is not directly interpretable. This man page aims to explain how to interpret the parameters and transform them into useful information.

Details

Parametrisation works differently for the scale parameters of the half-normals and for the mixture proportions.

The scale parameters of the half-normals (or their constituent parameters in the case of a covariate model) are given on the log scale.

The mixture proportions are transformed to a parametrisation that allows values to lie over all of the real line (see Miller and Thomas for details).

The parameter vector is made up of the scale parameters followed by the mixture parameters. In the non-covariate case the former is the length of the number of mixtures (`mix.terms`) and the latter is of length `mix.terms`-1. When the model has covariates the scale paramters are given as `mix.terms` intercepts followed by the covariate parameters.

The function `getpars` transforms the parameters (`$pars` element) in a `ds.mixture` object to a named list.

Calling `summary.ds.mixture` will show the mixture proportions.

The parameter `initialvalues` supplied to `fitmix`.

Author(s)

David L. Miller

plot.ds.mixture *Plot the results of a mixture model detection function fit.*

Description

Plots the detection function (or pdf) of a fitted mixture model detection function, optionally overlayed on a histogram of the observed data.

Usage

```
 ## S3 method for class 'ds.mixture'
plot(x, style = "", main = "",
   breaks = "Sturges", ylim = NULL, xlim = NULL,
   pdf = FALSE, plot.formula = NULL, hide.hist = FALSE,
   nomf = FALSE, x.axis = NULL, xlab = "Distance",
   ylab = NULL, ...)
```

Arguments

x	a ds.mixture object.
style	If set to "comp", composite plots of the detection function will be shown for the detection function (the averaged detection function in the covariate case).
main	(A vector of) title(s) for the plot(s). By default these are set by the function (and are fairly ugly but descriptive).
breaks	Breaks to be used for the histogram. This can be a vector of numbers or any of the permissable options used in hist. Defaults to "Sturges".
ylim	Used to manually set the y limit of the plot. Defaults to NULL.
xlim	Used to manually set the x limit of the plot. Defaults to NULL.
pdf	Should the pdf be plotted rather than detection function be plotted? Only really useful with point transect data. Defaults to FALSE.
plot.formula	Formula of covariates to be plotted. Defaults to NULL, which plots all covariates. No effect with non-covariate models.
hide.hist	Should the histogram be hidden, leaving only the detection function (or pdf) to be plotted? Defaults to FALSE.
nomf	Should the mfrow value be altered? Useful when creating custom plots for publication. Defaults to FALSE (yes, change the mfrow value).
x.axis	Set the x axis labels. Again, useful for publication plots. Defaults to NULL, which uses the default R values.
xlab	Label for the x axis.
ylab	Label for the y axis.
...	not used at the moment

Value

a plot!

Details

For covariate models, all the levels of factor variables are plotted or the 25, 50 and 75th percentiles of continuous variables are plotted averaged over the values of the other covariates.

Author(s)

David L, Miller

Examples

```
library(mmds)
set.seed(0)
## simulate some line transect data from a 2 point mixture
sim.dat<-sim.mix(c(-0.223,-1.897,inv.reparam.pi(0.3)),2,100,1)
## fit the model
fit.sim.dat<-fitmix(sim.dat,1,2)
## plot
plot(fit.sim.dat)
```

`reparam.pi` *Reparametrize mixture proportions*

Description

Re-parameterise the mixture proportions so that when there is more than a 2-point mixture, the proportions sum to 1.

Usage

```
reparam.pi(thetas)
```

Arguments

`thetas`	Mixture proportions in their parametrisation for optimization.

Value

a vector of parameters

Notes

See Miller and Thomas for information on exactly how these are calculated. Thanks go to David Borchers for proposing the trick.

Author(s)

David L. Miller

References

Miller, D.L. and L. Thomas (in prep.). Mixture model distance sampling detection functions.

See Also

inv.reparam.pi

Examples

```
library(mmds)
reparam.pi(inv.reparam.pi(0.3))
reparam.pi(inv.reparam.pi(c(0.3,0.4,0.1),lastpar=TRUE))
```

`sim.mix` *Simulate data from a mixture model detection function*

Description

Simulate data from a (line or point transect) mixture model detection function with or without covariates using rejection sampling.

Usage

```
sim.mix(pars, mix.terms, n, width, zdim = 0, z = NULL,
  pt = FALSE, showit = FALSE)
```

Arguments

`pars`	Parameters of the model to fit. See `mmds.pars` for details.
`mix.terms`	Number of mixture components.
`n`	Number of data to generate.
`width`	Truncation distance.
`zdim`	Number of columns of `z`. Defaults to 0.
`z`	Covariate data. Defaults to NULL. See details for more information.
`pt`	Should point transect data be generated? Defaults to FALSE.
`showit`	Print the acceptance rate. Defaults to FALSE.

Details

This routine uses rejection sampling, so may be rather slow of large sample sizes. Direct sampling will be available soon.

Value

a `data.frame` with the following columns:

observed	Whether the object was observed, always `n` 1s. Kept for `mmds` compatability.
object	Object identifier, numbered 1 to `n`. Kept for `mmds` compatability.
distance	Observed distances.
	Then follows as many columns as there are columns as `z`, named as in `z`.

Author(s)

David L. Miller

Examples

```
library(mmds)
set.seed(0)
## simulate some line transect data from a 2 point mixture
sim.dat<-sim.mix(c(-0.223,-1.897,inv.reparam.pi(0.3)),2,100,1)
hist(sim.dat$distance)
```

---

step.ds.mixture *Stepwise selection of mixture components*

Description

Uses AIC to select the number of mixture components.

Usage

```
step.ds.mixture(ds.object, max.terms = 4)
```

Arguments

`ds.object`	`ds.mixture` object.
`max.terms`	Maximum number of mixture components to attempt to fit. Default 4.

Value

a `ds.mixture` model object of the best (AIC-wise) model.

Details

This routine is most useful during model building. Setting up a basic 1-point mixture model and then running this on the object returns a model with the lowest AIC.

Progress will be printed to the screen.

Author(s)

David L. Miller

Examples

```
library(mmds)
set.seed(0)
## simulate some line transect data from a 2 point mixture
sim.dat<-sim.mix(c(-0.223,-1.897,inv.reparam.pi(0.3)),2,100,1)
## fit the model
fit.sim.dat.1<-fitmix(sim.dat,1,1)
## find best AIC model
step.ds.mixture(fit.sim.dat.1)
```

---

summary.ds.mixture *Summarize a ds.mixture object*

Description

Summarize a ds.mixture object. The function provides information on parameter estimates, estimates of the abundance in the covered area and the average detectability and their respective standard errors and coefficients of variation.

Usage

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

Arguments

object	A fitted mixture model detection function object.
...	Anything, but it will be ignored.

Value

a summary of a ds.mixture object.

Author(s)

David L. Miller

References

Miller, D.L. and L. Thomas (in prep.). Mixture model distance sampling detection functions.

Examples

```
library(mmds)
set.seed(0)
## simulate some line transect data from a 2 point mixture
sim.dat<-sim.mix(c(-0.223,-1.897,inv.reparam.pi(0.3)),2,100,1)
## fit the model
fit.sim.dat<-fitmix(sim.dat,1,2)
## what happened?
summary(fit.sim.dat)
```

Index