The bbmle Package

October 23, 2007

Title  Tools for general maximum likelihood estimation
Version  0.8.1
Date  2007/10/12
Author  Ben Bolker <bolker@zoo.ufl.edu>, based on stats4 by the R Development Core Team
Maintainer  Ben Bolker <bolker@zoo.ufl.edu>
Depends  R (>= 2.0.0), methods, stats, graphics, nlme
Suggests  aod, emdbook, Hmisc
Description  Methods and functions for fitting maximum likelihood models in R. This package modifies and extends the mle classes in the stats4 package.
License  GPL

R topics documented:

BIC-methods ................................................................. 2
ICtab ................................................................. 3
get.mnames ............................................................. 4
mle2-class ............................................................. 5
mle ................................................................. 6
mle2.options ........................................................... 8
namedrop ............................................................. 9
parnames ............................................................. 10
profile.mle2-class ...................................................... 11
relist2 ............................................................. 12
slice ............................................................. 12
slice.mle2-class ...................................................... 13
strwrapx ........................................................... 14
summary.mle2-class ................................................... 15

Index  16
Description

Various functions for likelihood-based and information-theoretic model selection of likelihood models.

Usage

```r
## S4 method for signature 'ANY, mle2, logLik':
BIC(object,...)
```

Arguments

- `object`: A `logLik` or `mle2` object.
- `...`: An optional list of additional `logLik` or `mle2` objects (fitted to the same data set). If the number of attributes has not been included as an attribute of the fit or of the log-likelihood, it can specified as `nobs` in this list.

Details

Further arguments to `BIC` can be specified in the `...` list: `delta` (logical) specifies whether to include a column for delta-BIC in the output.

Value

A table of the BIC values, degrees of freedom, and possibly delta-BIC values relative to the minimum-BIC model.

Methods

- `logLik` signature(object = "mle2") : Extract maximized log-likelihood.
- `AIC` signature(object = "mle2") : Calculate Akaike Information Criterion.
- `AICc` signature(object = "mle2") : Calculate small-sample corrected Akaike Information Criterion.
- `BIC` signature(object = "mle2") : Calculate Bayesian (Schwarz) Information Criterion.
- `BIC` signature(object = "logLik") : Calculate Bayesian (Schwarz) Information Criterion.
- `BIC` signature(object = "ANY") : Calculate Bayesian (Schwarz) Information Criterion.
- `anova` signature(object="mle2") : Likelihood Ratio Test comparision of different models.

Note

This is implemented in an ugly way and could probably be improved!
Examples

```r
x <- 0:10
y <- c(26, 17, 13, 12, 20, 5, 9, 8, 5, 4, 8)
(fit <- mle2(y~dpois(lambda=xmax/(1+x/xhalf)),
    start=list(xmax=25,xhalf=3)))
(fit2 <- mle2(y~dpois(lambda=(x+1)*slope),
    start=list(slope=1)))
BIC(fit,nobs=length(x))
BIC(fit,fit2,nobs=length(x))
```

ICtab

**Compute table of information criteria and auxiliary info**

Description

Computes information criteria for a series of models, optionally giving information about weights, differences between ICs, etc.

Usage

```r
ICtab(..., type=c("AIC","BIC","AICc"),
    weights = FALSE, delta = FALSE, sort = FALSE,
    nobs, dispersion = 1, mnames, k = 2)
AICtab(...)
BICtab(...)
AICctab(...)
## S3 method for class 'ICtab':
print(x,...)
```

Arguments

- `...`: a list of (logLik or?) mle objects; in the case of AICtab etc., could also include other arguments to ICtab
- `type`: specify information criterion to use
- `weights`: (logical) compute IC weights?
- `delta`: (logical) compute differences among ICs?
- `sort`: (logical) sort ICs in increasing order?
- `nobs`: (logical) number of observations: required for type="BIC" or type="AICc" unless objects have an "nobs" attribute
- `dispersion`: (stub) overdispersion estimate, for computing qAIC
- `mnames`: names for table rows: defaults to names of objects passed
- `k`: penalty term (largely unused)
- `x`: an ICtab object
4

get.mnames

Value

A data frame containing:

IC  information criterion
df  degrees of freedom/number of parameters
dIC difference in IC from minimum-IC model
weights  \(\exp(-dIC/2)/\text{sum}(\exp(-dIC/2))\)

Note

The print method uses sensible defaults; all ICs are rounded to the nearest 0.1, and IC weights are printed using \texttt{format.pval} to print an inequality for values <0.001

Author(s)

Ben Bolker

References

Burnham and Anderson 2002

---

**get.mnames**  
extract model names

Description

given a list of models, extract the names (or "model n")

Usage

\texttt{get.mnames(Call)}

Arguments

\texttt{Call}  
a function call (usually a list of models)

Value

a vector of model names

Author(s)

Ben Bolker
**mle2-class**

Class "mle2". Result of Maximum Likelihood Estimation.

Description

This class encapsulates results of a generic maximum likelihood procedure.

Objects from the Class

Objects can be created by calls of the form `new("mle2", ...)`, but most often as the result of a call to `mle2`.

Slots

- **call**: (language) The call to `mle2`.
- **coef**: (numeric) Vector of estimated parameters.
- **fullcoef**: (numeric) Fixed and estimated parameters.
- **vcov**: (numeric matrix) Approximate variance-covariance matrix, based on the second derivative matrix at the MLE.
- **min**: (numeric) Minimum value of objective function = minimum negative log-likelihood.
- **details**: (list) Return value from `optim`.
- **minuslogl**: (function) The negative log-likelihood function.
- **method**: (character) The optimization method used.
- **formula**: (character) If a formula was specified, a character vector giving the formula and parameter specifications.

Methods

- **confint** signature(object = "mle2"): Confidence intervals from likelihood profiles.
- **profile** signature(fitted = "mle2"): Likelihood profile generation.
- **show** signature(object = "mle2"): Display object briefly.
- **summary** signature(object = "mle2"): Generate object summary.
- **update** signature(object = "mle2"): Update fit.
- **vcov** signature(object = "mle2"): Extract variance-covariance matrix.
- **plot** signature(object="profile.mle2,missing"): Plot profile.
mle

Maximum Likelihood Estimation

Description

Estimate parameters by the method of maximum likelihood.

Usage

mle2(minuslogl, start, method, optimizer,
       fixed = NULL, data=NULL, subset=NULL,
       default.start=TRUE, eval.only = FALSE, vecpar=FALSE,
       parameters=NULL, skip.hessian=FALSE,trace=FALSE,gr,...)
calc_mle2_function(formula,parameters,start,data=NULL,trace=FALSE)

Arguments

minuslogl  Function to calculate negative log-likelihood.
start      Named list. Initial values for optimizer.
method     Optimization method to use. See optim.
optimizer  Optimization function to use. (Stub.)
fixed      Named list. Parameter values to keep fixed during optimization.
data       list of data to pass to minuslogl
subset     logical vector for subsetting data (STUB)
default.start Logical: allow default values of minuslogl as starting values?
eval.only  Logical: return value of minuslogl(start) rather than optimizing
vecpar     Logical: is first argument a vector of all parameters? (For compatibility with optim.)
parameters List of linear models for parameters
gr         gradient function
...        Further arguments to pass to optimizer
formula    a formula for the likelihood (see Details)
trace      Logical: print parameter values tested?
skip.hessian Bypass Hessian calculation?
Details

The \texttt{optim} optimizer is used to find the minimum of the negative log-likelihood. An approximate covariance matrix for the parameters is obtained by inverting the Hessian matrix at the optimum.

The \texttt{minuslogl} argument can also specify a formula, rather than an objective function, of the form \( x \sim \text{ddistn}(\text{param1}, \ldots, \text{paramn}) \). In this case \texttt{ddistn} is taken to be a probability or density function, which must have (literally) \( x \) as its first argument (although this argument may be interpreted as a matrix of multivariate responses) and which must have a \texttt{log} argument that can be used to specify the log-probability or log-probability-density is required. If a formula is specified, then \texttt{parameters} can contain a list of linear models for the parameters.

If a formula is given and non-trivial linear models are given in \texttt{parameters} for some of the variables, then model matrices will be generated using \texttt{model.matrix}. \texttt{start} can either be an exhaustive list of starting values (in the order given by \texttt{model.matrix}) or values can be given just for the higher-level parameters: in this case, all of the additional parameters generated by \texttt{model.matrix} will be given starting values of zero.

The \texttt{trace} argument applies only when a formula is specified. If you specify a function, you can build in your own \texttt{print()} or \texttt{cat()} statement to trace its progress. (You can also specify a value for \texttt{trace} as part of a \texttt{control} list for \texttt{optim}: see \texttt{optim}.)

The \texttt{skip.hessian} argument is useful if the function is crashing with a "non-finite finite difference value" error when trying to evaluate the Hessian, but will preclude many subsequent confidence interval calculations. (You will know the Hessian is failing if you use \texttt{method="Nelder-Mead"} and still get a finite-difference error.)

If convergence fails, see \texttt{optim} for the meanings of the error codes.

Value

An object of class "\texttt{mle2}".

Note

Note that the \texttt{minuslogl} function should return \(-\log L\) (not \(-2 \log L\)). It is the user’s responsibility to ensure that the likelihood is correct, and that asymptotic likelihood inference is valid (e.g. that there are "enough" data and that the estimated parameter values do not lie on the boundary of the feasible parameter space).

See Also

\texttt{mle2-class}

Examples

\begin{verbatim}
x <- 0:10
y <- c(26, 17, 13, 12, 20, 5, 9, 8, 5, 4, 8)
LL <- function(ymax=15, xhalf=6)
  -sum(stats::dpois(y, lambda=ymax/(1+x/xhalf), log=TRUE))
## uses default parameters of LL
(fit <- mle2(LL))
mle2(LL, fixed=list(xhalf=6))
\end{verbatim}
mle2.options

Options for maximum likelihood estimation

Description

Query or set MLE parameters

Usage

mle2.options(...)

Arguments

... names of arguments to query, or a list of values to set
Details

optim.method name of optimization method (see \texttt{optim} for choices)

confint name of confidence-interval: choices are "spline", "uniroot", "hessian" corresponding to spline inversion, attempt to find best answer via uniroot, information-matrix approximation

optimizer optimization function to use (only choice at present is "optim")

Value

Values of queried parameters, or (invisibly) the full list of parameters

See Also

\texttt{mle2-class}

\begin{verbatim}
\begin{verbatim}
\texttt{namedrop} \hspace{1cm} \textit{drop unneeded names from list elements}
\end{verbatim}
\end{verbatim}

Description

goes through a list (containing a combination of single- and multiple-element vectors) and removes redundant names that will make trouble for \texttt{mle}

Usage

\texttt{namedrop(x)}

Arguments

\begin{itemize}
  \item \texttt{x} a list of named or unnamed, typically numeric, vectors
\end{itemize}

Details

examines each element of \texttt{x}. If the element has length one and is a named vector, the name is removed; if \texttt{length(x)} is greater than 1, but all the names are the same, the vector is renamed

Value

the original list, with names removed/added

Author(s)

Ben Bolker
Examples

\[
x = \text{list}(a=c(a=1),b=c(d=1,d=2),c=c(a=1,b=2,c=3))
\]
\[
\text{names(unlist(namedrop(x)))}
\]
\[
\text{names(unlist(namedrop(x)))}
\]

generate parameter names

Description

Gets and sets the "parnames" attribute on a negative log-likelihood function

Usage

\[
\text{parnames(obj)}
\]
\[
\text{parnames(obj) <- value}
\]

Arguments

\[
\text{obj} \quad \text{a negative log-likelihood function}
\]
\[
\text{value} \quad \text{a character vector of parameter names}
\]

Details

The `parnames` attribute is used by `mle2()` when the negative log-likelihood function takes a parameter vector, rather than a list of parameters; this allows users to use the same objective function for `optim()` and `mle2()`

Value

Returns the `parnames` attribute (a character vector of parameter names) or sets it.

Author(s)

Ben Bolker

Examples

\[
x <- 1:5
\]
\[
\text{set.seed(1001)}
\]
\[
y <- \text{rbinom(5,prob=x/(1+x),size=10)}
\]
\[
mfun <- \text{function(p) \{}
   \text{a <- p[1]}
   \text{b <- p[2]}
   \text{-sum(dbinom(y,prob=a*x/(b+x),size=10,log=TRUE))}
\}
\]
\[
\text{optim(fn=mfun,par=c(1,1))}
\]
\[
\text{parnames(mfun) <- c("a","b")}
\]
\[
\text{mle2(minuslogl=mfun,start=c(a=1,b=1),method="Nelder-Mead")}
\]
profile.mle2-class

Class "profile.mle2"; Profiling information for "mle2" object

Description

Likelihood profiles along each parameter of likelihood function

Objects from the Class

Objects can be created by calls of the form new("profile.mle2", ...), but most often by invoking profile on an "mle2" object.

Slots

profile: Object of class "list". List of profiles, one for each requested parameter. Each profile is a data frame with the first column called z being the signed square root of the -2 log likelihood ratio, and the others being the parameters with names prefixed by par.vals.

summary: Object of class "summary.mle2". Summary of object being profiled.

Methods

confint signature(object = "profile.mle2"): Use profile to generate approximate confidence intervals for parameters.

plot signature(x = "profile.mle2", y = "missing"): Plot profiles for each parameter.

summary signature(x = "profile.mle2"): Plot profiles for each parameter.

show signature(object = "profile.mle2"): Show object.

See Also

mle2, mle2-class, summary.mle2-class

Examples
relist2  
*reconstruct the structure of a list*

**Description**

reshapes a vector according to a list template

**Usage**

relist2(v, l)

**Arguments**

- `v`: vector, probably numeric, of values to reshape
- `l`: template list giving structure

**Details**

attempts to coerce `v` into a list with the same structure and names as `l`

**Value**

a list with values corresponding to `v` and structure corresponding to `l`

**Author(s)**

Ben Bolker

**Examples**

```r
l = list(b=1,c=2:5,d=matrix(1:4,nrow=2))
relist2(1:9,l)
```

---

slice  
*Calculate likelihood "slices"*

**Description**

Computes a cross-section of a multi-dimensional likelihood surface

**Usage**

slice(fitted, ...)

slice.mle2-class

Arguments

  fitted a fitted object of class mle

  ... other arguments: which etc.

Value

  still a stub.

Note

  Slices provide a lighter-weight way to explore likelihood surfaces than profiles, since they vary a single parameter rather than

Author(s)

  Ben Bolker

See Also

  profile

slice.mle2-class likelihood-surface slices

Description

  evaluations of log-likelihood along transects in parameter space

Objects from the Class

  Objects can be created by calls of the form new("slice.mle2", ...). The objects are similar to likelihood profiles, but don’t involve any optimization with respect to the other parameters.

Slots

  profile: Object of class "list". List of slices, one for each requested parameter. Each slice is a data frame with the first column called z being the signed square root of the -2 log likelihood ratio, and the others being the parameters with names prefixed by par.vals.

  summary: Object of class "summary.mle2". Summary of object being profiled.

Methods

  plot signature(x = "profile.mle2", y = "missing"): Plot profiles for each parameter.

See Also

  profile.mle2-class
\textit{strwrapx} \hspace{1cm} \textit{Wrap strings at white space and + symbols}

\textbf{Description}

Extended (hacked) version of \texttt{strwrap}: wraps a string at whitespace and plus symbols

\textbf{Usage}

\begin{verbatim}
strwrapx(x, width = 0.9 * getOption("width"), indent = 0, exdent = 0, prefix = "", simplify = TRUE, parsplit = "\[ \t\n\] *
", wordsplit = "\[ \t\n\] ")
\end{verbatim}

\textbf{Arguments}

- \texttt{x}: a character vector, or an object which can be converted to a character vector by \texttt{as.character}.
- \texttt{width}: a positive integer giving the target column for wrapping lines in the output.
- \texttt{indent}: a non-negative integer giving the indentation of the first line in a paragraph.
- \texttt{exdent}: a non-negative integer specifying the indentation of subsequent lines in paragraphs.
- \texttt{prefix}: a character string to be used as prefix for each line.
- \texttt{simplify}: a logical. If \texttt{TRUE}, the result is a single character vector of line text; otherwise, it is a list of the same length as \texttt{x} the elements of which are character vectors of line text obtained from the corresponding element of \texttt{x}. (Hence, the result in the former case is obtained by unlisting that of the latter.)
- \texttt{parsplit}: Regular expression describing how to split paragraphs
- \texttt{wordsplit}: Regular expression describing how to split words

\textbf{Details}

Whitespace in the input is destroyed. Double spaces after periods (thought as representing sentence ends) are preserved. Currently, possible sentence ends at line breaks are not considered specially.

Indentation is relative to the number of characters in the prefix string.

\textbf{Examples}

\begin{verbatim}
## Read in file 'THANKS'.
x <- paste(readLines(file.path(R.home("doc"), "THANKS")), collapse = "\n")
## Split into paragraphs and remove the first three ones
x <- unlist(strsplit(x, "\n[ \t\n\]"))[-(1:3)]
## Join the rest
x <- paste(x, collapse = "\n\n")
## Now for some fun:
\end{verbatim}
## summary.mle2-class

### Class "summary.mle2", summary of "mle2" objects

#### Description

Extract of "mle2" object

#### Objects from the Class

Objects can be created by calls of the form `new("summary.mle2", ...),` but most often by invoking `summary` on an "mle2" object. They contain values meant for printing by `show`.

#### Slots

- **call**: Object of class "language" The call that generated the "mle2" object.
- **coef**: Object of class "matrix". Estimated coefficients and standard errors
- **m2logL**: Object of class "numeric". Minus twice the log likelihood.

#### Methods

- **show** signature(object = "summary.mle2"): Pretty-prints object
- **coef** signature(object = "summary.mle2"): Extracts the contents of the coef slot

#### See Also

`summary.mle2, mle2-class`
Index

*Topic character
  strwrapx, 14

*Topic classes
  mle2-class, 5
  profile.mle2-class, 11
  slice.mle2-class, 13
  summary.mle2-class, 15

*Topic methods
  BIC-methods, 1

*Topic misc
  get.mnames, 4
  ICtab, 3
  namedrop, 9
  parnames, 10
  relist2, 12
  slice, 12

*Topic models
  mle, 6
  mle2, 5–7, 9
  mle2-class, 7, 9, 11, 15
  mle2-class, 5
  mle2.options, 8
  namedrop, 9
  optim, 5–7, 9
  parnames, 10
  parnames<-(parnames), 10
  plot, profile.mle2, missing-method
    (profile.mle2-class), 11
  plot, profile.mle2-method
    (profile.mle2-class), 11
  print.ICtab(ICtab), 3
  BIC, mle2-method (BIC-methods), 1
  BIC-methods, 1
  BICtab(ICtab), 3
  calc_mle2_function(mle), 6
  coef, mle2-method (mle2-class), 5
  coef, summary.mle2-method
    (summary.mle2-class), 15
  confint, mle2-method (mle2-class), 15
  confint, profile.mle2-method
    (profile.mle2-class), 11
  format.pval, 4
  get.mnames, 4
  ICtab, 3
  logLik, mle2-method (BIC-methods), 1
  logLik-methods (BIC-methods), 1
  mle, 6
  mle2, 5, 11, 15
  mle2(mle), 6
  mle2-class, 7, 9, 11, 15
  mle2-class, 5
  mle2.options, 8
  namedrop, 9
  optim, 5–7, 9
  parnames, 10
  parnames<-(parnames), 10
  plot, profile.mle2, missing-method
    (profile.mle2-class), 11
  plot, profile.mle2-method
    (profile.mle2-class), 11
  print.ICtab(ICtab), 3
INDEX

profile, 13
profile.mle2-method(mle2-class), 5
profile.mle2-class, 13
profile.mle2-class, 11
relist2, 12
show,mle2-method(mle2-class), 5
show,profile.mle2-method
(profile.mle2-class), 11
show,summary.mle2-method
(summary.mle2-class), 15
slice, 12
slice,mle2-method(mle2-class), 5
slice.mle2-class, 13
strwrapx, 14
summary, 15
summary,mle2-method(mle2-class), 5
summary.mle2-class, 11
summary.mle2-class, 15
update,mle2-method(mle2-class), 5
vcov,mle2-method(mle2-class), 5