Package: glmulti (via r-universe)

October 28, 2026

Version 1.0.9

Date 2026-10-25

Title Model Selection and Multimodel Inference Made Easy

Author Vincent Calcagno [aut, cre]

Maintainer Vincent Calcagno <vincent.calcagno@inrae.fr>

SystemRequirements Java (>= 5.0)

Depends R (>= 2.13.0), rJava (>= 0.5-0), methods, leaps

Suggests MASS, aod, lme4, pscl, survival, nnet

Description Automated model selection and model-averaging. Provides a wrapper for glm and other functions, automatically generating all possible models (under constraints set by the user) with the specified response and explanatory variables, and finding the best models in terms of some Information Criterion (AIC, AICc or BIC). Can handle very large numbers of candidate models. Features a Genetic Algorithm to find the best models when an exhaustive screening of the candidates is not feasible.

License GPL (>= 2)

NeedsCompilation no

Config/pak/sysreqs default-jdk

Repository https://cran.r-universe.dev

Date/Publication 2026-10-27 20:10:02 UTC

RemoteUrl https://github.com/cran/glmulti

RemoteRef HEAD

RemoteSha 130ed1e57c2550bfd76c88f12668e39796ea0418

Contents

aic 2
aic-methods 3
aicc-methods 3

bic-methods . . . 3
coef.glmulti . . . 4
consensus . . . 5
consensus-methods . . . 6
getfit . . . 7
getfit-methods . . . 8
glmulti . . . 8
glmulti-class . . . 12
glmulti-methods . . . 13
qaic-methods . . . 14
qaicc-methods . . . 14
summary.glmulti . . . 14
weightable . . . 16
weightable-methods . . . 17
write . . . 17
write-methods . . . 18

Index **19**

`aic` *Computing an IC from a fitted model object*

Description

These functions are used by `glmulti` to compute Information Criteria from a fitted model object. They are S4 generics, currently with methods for `glm`/`lm` objects. You may define your own methods if needed, to support other fitting functions or to use other IC.

Usage

```
aic(object, ...)
aicc(object, ...)
bic(object, ...)
qaic(object, ...)
qaicc(object, ...)
```

Arguments

`object`	A fitted model object.
`...`	Further arguments that may be useful for future methods.

Details

When using `qaic` or `qaicc`, you will need to provide an estimate of the inflation factor c. This is done through the global variable `glmultiqaiccvalue`. Set this varaible equal to your estimated c before running `glmulti`. Otherwise it will stop with a warning.

Value

The IC value of the model.

Author(s)

Vincent Calcagno, McGill University

See Also

glmulti

`aic-methods` *Methods for Function aic*

Description

Used to compute the AIC value for several classes of models.

Methods

object = "ANY" Default method for e.g. `glm` and `lm`

`aicc-methods` *Methods for Function aicc*

Description

Used to compute the AICc value for several classes of models.

Methods

object = "ANY" Default method for e.g. `glm` and `lm`

`bic-methods` *Methods for Function bic*

Description

Used to compute the BIC value for several classes of models.

Methods

object = "ANY" Default method for e.g. `glm` and `lm`

`coef.glmulti` *Model averaging and multimodel inference with glmulti*

Description

These functions, applied on a `glmulti` object, produce model-averaged estimates, unconditional confidence intervals, and predictions from the models in the confidence set (or a subset of them). They are equivalents of the standard `coef` and `predict` for single models.

Usage

```
# S3 coef method for class 'glmulti'
## S3 method for class 'glmulti'
coef(object, select="all", varweighting="Buckland", icmethod="Lukacs", alphaIC=0.05, ...)

# S3 predict method for class 'glmulti'
## S3 method for class 'glmulti'
predict(object, select="all", newdata=NA, se.fit=FALSE,
 varweighting="Buckland", icmethod="Lukacs", alphaIC=0.05,  ...)
```

Arguments

`object`	an object of class `glmulti`
`select`	A specification of which models should be used for inference. By default all models are used, see below.
`varweighting`	The method to be used to compute the unconditional variance. "Buckland" (the default) (implements the approach presented in Buckland et al. 1997. "Johnson" implements a slightly different approach recommended in Johnson & Omland 2004 and proposed at page 235 in Burnham & Anderson 2002. The latter results in slightly bigger estimates of the unconditional variance of model coefficients.
`icmethod`	Method to construct confidence intervals. One of "Standard", "Burnham" or "Lukacs". The three methods differ in their use (or not) of degrees of freedom.
`newdata`	New data.frame of data for which to predict values
`se.fit`	Whether to return unconditional variances and confidence intervals associated with predicted values
`alphaIC`	The alpha risk when building the confidence intervals
`...`	Further arguments to single-model `coef` or `predict`

Details

select can be used to specify which models should be used for inference. By default all are used. If specifying an integer value *x*, only the *x* best models are used. If a numeric value is provided, if it less than one, models that sum up to *x%* of evidence weight are used. If it more than one, models within *x* IC units from the best model are used.

Value

`coef` returns a data.frame with model-averaged estimates of the different parameters in the models, as well as their unconditional variance, importance, and confidence interval according to one of three methods: "Standard" simply assumes a Normal distribution of the estimator (Buckland 1997), "Lukacs" assumes a Student distribution with degrees of freedom taken to be averaged across models (see Lukacs et al. 2010), and "Burnham" is a more sophisticated Student-based method proposed by Burnham & Anderson 2002.

`predict` returns a list of three elements: the multi-model predictions, their variability (unconditional variance and confidence interval, if `se.fit=T`), and the number of NA predicted values that were treated as zeros when averaging models.

Author(s)

Vincent Calcagno, McGill University

References

Buckland et al. 1997. Model selection: an integral part of inference. Biometrics. Burnham & Anderson. 2002. Model Selection and Multimodel Inference. An Information Theoretic Practical Approach. Johnson & Omland. 2004. Model selection in ecology and evolution. TREE. Lukacs et al. 2010..Model selection bias and Freedman's paradox. Annals of the Institute of Statistical Mathematics.

See Also

`glmulti`

`consensus` *Takes a consensus of several glmulti objects*

Description

The function takes a list of `glmulti` objects and returns a new `glmulti` object. This object contains the best models found in all passed objects, with duplicates removed. Useful to bring back together the different parts of a parallelized exhaustive screen, or to make the consensus of several replicate genetic algorithms.

Usage

```
consensus(xs, confsetsize = NA, ...)
```

Arguments

xs	A list containing glmulti objects, or filenames from which glmulti objects can be read on the disk (typically glmulti objects that have been saved using write with file="filename|object").
confsetsize	The number of best models to be included in the consensus object. If NA, all available models are included.
...	Further arguments, allowing to write custom S4 methods for this generic.

Details

The name of the returned object is the name of the first object passed, with "consensus-" prepended. The params slot of the returned object is taken from the first object too. Note that if not all glmulti objects in xs do contain fitted model objects, then no model object will be included in the returned glmulti object.

Value

A standard object of class glmulti

Author(s)

Vincent Calcagno, McGill University

References

Calcagno & de Mazancourt 2010 J. Stat. Soft. v34 i12. See http://www.jstatsoft.org/v34/i12

See Also

glmulti

---

consensus-methods *Consensus method for glmulti objects.*

Description

Concatenates several glmulti objects and makes a consensus of them.

Methods

xs = "list" a list of glmulti objects to make a consensus of. Can include actual objects or filenames pointing to objects written on the disk (typically glmulti objects that have been saved using write with file="filename|object").

getfit *Accessing coefficients of a fitted model object*

Description

Generic S4 function used to access the coefficients (and their standard error) from a fitted model. It is used by `coef.glmulti`.

Usage

```
getfit(object, ...)
```

Arguments

`object`	A fitted model object, of class `glm`, `lm` or `glm.nb`
`...`	Further arguments that may be required by some methods of this generic.

Details

Define your own method for this generic when using `glmulti` with some custom fitting function.

Value

A data.frame, with as rows the coefficients that are fitted in the model, and three columns: the first with the estimated coefficients, the second with the associated standard errors, and the third with degrees of freedom.

Author(s)

Vincent Calcagno, McGill University

References

Calcagno & de Mazancourt 2010 J. Stat. Soft. v34 i12. See http://www.jstatsoft.org/v34/i12

See Also

`glmulti`

getfit-methods *Methods for Function getfit*

Description

Used by `coef.glmulti` to access the contents of fitted model objects of different classes.

Methods

object = "ANY" Default method, for e.g. `glm` objects.

object = "coxph" Method for `coxph` objects.

object = "coxph.null" Method for `coxph.null` objects.

glmulti *Automated model selection and multimodel inference with (G)LMs*

Description

`glmulti` finds what are the `n` best models (the confidence set of models) among all possible models (the candidate set, as specified by the user). Models are fitted with the specified fitting function (default is `glm`) and are ranked with the specified Information Criterion (default is `aicc`). The best models are found either through exhaustive screening of the candidates, or using a genetic algorithm, which allows very large candidate sets to be adressed. The output can be used for model selection, variable selection, and multimodel inference.

Usage

```
#  glmulti S4 generic
glmulti(y, xr, data,
 exclude = c(),
 name = "glmulti.analysis",
 intercept = TRUE,
  marginality = FALSE,
  bunch=30,
  chunk = 1,
  chunks = 1,
  level = 2,
  minsize = 0,
  maxsize = -1,
  minK = 0,
  maxK = -1,
  method = "h",
  crit = "aic",
  confsetsize = 100,
```

```
  popsize = 100,
  mutrate = 10^-3,
  sexrate = 0.1,
  imm = 0.3,
  plotty = TRUE,
  report = TRUE,
  deltaM = 0.05,
  deltaB = 0.05,
  conseq = 5,
  fitfunction = "glm",
  resumefile = "id",
  includeobjects=TRUE,
  ...)
```

Arguments

y	A formula, character string, or fitted model (of class lm or glm) specifying the response variable and the terms (main effects and/or interactions) to be used in the candidate models (e.g. height~age*sex+mass). Alternatively, a character string naming the variable to be used as response (e.g. "height") (in which case the names of the predictors must be passed through the xr argument) Alternatively, a custom list of (fitted) model objects can also be passed (can be convenient for small candidate sets).
xr	An optional character array specifying the variables (categorical or quantitative) to be used as predictors, e.g. c("age", "height" , "mass")
exclude	Optional character vector naming terms (main effects or interactions) to be excluded from the candidate models, e.g. c("mass:height")
intercept	Whether to include an intercept in the candidate models or not.
level	If 1, only main effects (terms of order 1) are used to build the candidate set. If 2, pairwise interactions are also used (higher order interactions are currently ignored)
data	A data.frame containing the data. If not specified, `glmulti` will try to find the data in the environment of the formula, from the fitted model passed as y argument, or from the global environment.
name	The name of this `glmulti` analysis. Optional.
marginality	Whether to apply the marginality rule or not. If TRUE, only marginal models will be considered.
minsize	This sets a constraint on candidate models. Minimal number of TERMS (main effects or interactions) to be included in candidate models (negative = no constraint)
maxsize	This sets a constraint on candidate models. Maximal number of TERMS to be included in candidate models (negative = no constraint)
minK	This sets a constraint on candidate models. Minimal complexity of candidate models (negative = no constraint)

maxK	This sets a constraint on candidate models. Maximal complexity of candidate models (negative = no constraint)
method	The method to be used to explore the candidate set of models. If "h" an exhaustive screening is undertaken. If "g" the genetic algorithm is employed (recommended for large candidate sets). If "l", a very fast exhaustive branch-and-bound algorithm is used. Package leaps must then be loaded, and this can only be applied to linear models with covariates and no interactions. If "d", a simple summary of the candidate set is printed, including the number of candidate models.
crit	The Information Criterion to be used. This should be a function that accepts a fitted model as first argument. Default is the original Akaike IC (aic). Other provided functions are the Bayes IC (bic), the small-sample corrected AIC (aicc) and QAIC/QAICc (qaic and qaicc).
fitfunction	The fitting function to be used. Any function similar to glm can be used. See Examples
confsetsize	The number of models to be looked for, i.e. the size of the returned confidence set.
plotty	Whether to plot the progress of the IC profile when running.
report	Whether to report about the progress at run time.
bunch	The number of model formulas to be returned (to be fitted) at each call to the enumerator. Exhaustive screening only.
chunk	When using an exhaustive screening approach, it can be splitted in several parts to take advantage of multiple CPUs. chunk is an integer specifying which part the current call should perform.
chunks	When splitting an exhaustive screening, the total number of parts the task should be divided into. For example, with a quad-core processor, 4 may be useful. Use consensus to bring back the pieces into a single object.
popsize	The population size for the genetic algorithm
mutrate	The per locus (i.e. per term) mutation rate for genetic algorithm, between 0 and 1
sexrate	The rate of sexual reproduction for the genetic algorithm, between 0 and 1
imm	The rate of immigration for the genetic algorithm, between 0 and 1
deltaM	The target change in mean IC (defines the stop rules for the genetic algorithm)
deltaB	The target change in best IC (defines the stop rules for the genetic algorithm)
conseq	The target successive number of times with no improvement (i.e. target changes have been attained) (defines the stop rule for the GA). The greater it is, the more stringent the stop rule.
resumefile	When resuming an analysis (method="r"), the name of the files from which to resume. Default uses the same as name
includeobjects	Whether or not to include fiited models as objects. This makes coef and predict faster and is very convenient, but can be turned off in case fitted models are very large or are not to be used after.
...	Further arguments to be passed to the fitting function. E.g. maxit=50 or family=binomial

Details

`glmulti` is defined as a S4 function. It acts as a frontend that calls background compiled functions (contained if archive glmulti.jar). Running the function therefore requires a Java Running Environment, and package `rJava`. A thorough description of this function and package can be found in the article by Calcagno and de Mazancourt (see References). `print.glmulti` and `summary.glmulti` are S3 methods which provide a synthesis of `glmulti` analyses.

NOTE: When calling `glmulti` with a model object as y, only the formula will be extracted from the object. This means that optional arguments to the fitting function (e.g. family or maxit) will NOT be extracted. These arguments should be passed to `glmulti` through the ....

Value

An object of class `glmulti` is returned. It is a S4 object with several slots containing relevant data for model selection and beyond.

Several standard S3 functions are provided to help access the content of this object.

Several `glmulti` objects can be shrunk to one using the function consensus. This is useful to get the best of several replicates (of the genetic algorithm) or to bring together the different parts of a splitted exhaustive screening. When running a genetic algorithm, two tiny `java` files (serialized objects) are also written to the disk at regular intervals. They can be used to resume the calculation (method="r") if it was interrupted for any reason. This can also be used to continue a GA with modified parameters (e.g. mutation rate).

Author(s)

Vincent Calcagno, McGill University, Canada

References

Buckland (1997) Model Selection: an Integral Part of Inference. Biometrics 10:41 Burnham & Anderson (2002) Model Selection and Multimodel Inference: an Information Theoretic Approach Calcagno & de Mazancourt 2010 J. Stat. Soft. v34 i12. See http://www.jstatsoft.org/v34/i12

See Also

`consensus`, `aic`, `weightable`, `summary.glmulti`, `coef.glmulti`, `step`

Examples

```
# See the document "glmulti.pdf" included in the package.
# It explains the general approach
# and shows how to use glmulti with mixed models from the lme4 package.
# Other examples:
# A. This shows how to do the same for zero-inflated poisson models
# we load the required package
library(pscl)
# a random vector of count data
round(runif(100, 0,20)*round(runif(100)))-> vy2
# dummy predictors
va = runif(100)
```

```
vb = runif(100)
# 1. The wrapper function
zeroinfl.glmulti=function(formula, data, inflate = "|1",...)  {
    zeroinfl(as.formula(paste(deparse(formula), inflate)),data=data,...)
}
# The default getfit and aicc method will work for zeroinfl objects, so no need to redefine them
# we can proceed directly
glmulti(vy2~va*vb,fitfunc=zeroinfl.glmulti,inflate="|1")->bab


# B. This shows how to include some terms in ALL the models
# As above, we just prepare a wrapper of the fitting function
glm.redefined = function(formula, data, always="", ...) {
glm(as.formula(paste(deparse(formula), always)), data=data, ...)
}
# we then use this fitting function in glmulti
glmulti(vy2~va,level=1,fitfunc=glm.redefined,always="+vb")-> bab
# va will be shuffled but vb is always included in the models

# this procedure allows support of arbitrarily any fitting function,
# or the use of sophisticated constraints on the model structure
```

`glmulti-class` *Class "glmulti"*

Description

Contains the results of a `glmulti` analysis.

Objects from the Class

Objects will never be created directly but through calls of `glmulti` or by applying `consensus` on a list of `glmulti` objects.

Slots

`name`**:** Object of class `"character"` : the name of the analysis.

`params`**:** Object of class `"list"` : parameter values used when calling `glmulti` to produce the object.

`nbmods`**:** Object of class `"integer"` : the number of models that have been found by `glmulti`.

`crits`**:** Object of class `"numeric"` : the IC values of the models found, in ascending order.

`K`**:** Object of class `"integer"` : for each model, its complexity, from best to worst model.

`formulas`**:** Object of class `"list"` : for each model, its formula, from best to worst.

`call`**:** Object of class `"call"` : the original `glmulti` call that produced this object.

`adi`**:** Object of class `"list"` : additional arguments that had been passed to the fitting function through `glmulti`.

`objects`**:** Object of class `"list"` : The list of fitted model objects for the confidence set, if `includeobjects=T`.

Author(s)

Vincent Calcagno, McGill University

References

Calcagno & de Mazancourt 2010 J. Stat. Soft. v34 i12. See http://www.jstatsoft.org/v34/i12

See Also

summary.glmulti, consensus, coef.glmulti

Examples

```
showClass("glmulti")
```

---

glmulti-methods *Methods for Function glmulti: different ways to call glmulti*

Description

glmulti finds what are the n best models (the confidence set of models) among all possible models (the candidate set, as specified by the user). Models are fitted with the specified fitting function (default is glm) and are ranked with the specified Information Criterion (default is aic). The best models are found either through exhaustive screening of the candidates or using a genetic algorithm, which allows very large candidate sets to be adressed. The output can be used for model selection, variable selection, and multimodel inference.

Methods

y = "ANY", xr = "ANY", data = "ANY", exclude = "ANY" This will stop with a warning that an improper call has been attempted.

y = "character", xr = "character", data = "ANY", exclude = "ANY" Calling glmulti by providing the names of the response variable and of the predictors as character strings. This is the original interface used in versions earlier than 0.6-1.

y = "character", xr = "missing", data = "ANY", exclude = "missing" Calling glmulti with a model formula represented as a character string. E.g. "u~c+x"

y = "formula", xr = "missing", data = "ANY", exclude = "missing" Calling glmulti with a model formula containing all the terms to be included in candidate models. E.g. u~c+x

y = "list", xr = "ANY", data = "ANY", exclude = "ANY" Calling glmulti on a list of (fitted) model objects. Models will not be refitted, but the information criteria will be computed and a regular glmulti object is returned.

y = "glm", xr = "missing", data = "ANY", exclude = "missing" Calling glmulti with a glm object from which the formula and other parameters will eb extracted.

y = "lm", xr = "missing", data = "ANY", exclude = "missing" Calling glmulti with a lm model from which the formula and other parameters will eb extracted..

y = "missing", xr = "ANY", data = "ANY", exclude = "ANY" If y is missing the glmulti version currently used is printed.

qaic-methods *Methods for Function qaic*

Description

Used to compute the QAIC value for several classes of models. Specify the estimated c value to be used through the "glmulti-cvalue" option.

Methods

object = "ANY" Default method for e.g. `glm` and `lm`

qaicc-methods *Methods for Function qaicc*

Description

Used to compute the QAICc value for several classes of models. Specify the estimated c value to be used through the "glmulti-cvalue" option.

Methods

object = "ANY" Default method, for e.g. `glm` and `lm`

summary.glmulti *Handling glmulti objects*

Description

These standard S3 functions can be applied on `glmulti` to print a short report, obtain a more detailed summary, or produce different types of graphics.

Usage

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

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


# S3 plot method for class 'glmulti'
```

```
## S3 method for class 'glmulti'
plot(x, type="p", ...)
```

Arguments

object	A glmulti object
x	An object of class glmulti
type	The type of graph to be produced. One of "p", "r", "s" or "w" (see below).
...	Further arguments.

Details

The name of the returned object is the name of the first object passed, with "consensus-" prepended. The params slot of the returned object is taken from the first object too.

Value

plot can be used to have a graphical representation of the results. Two types are proposed:

type="p" plots the IC profile (the IC values form the best to the worst model). A horizontal line delineates models that are less than 2 IC units away from the best model.

type="r" shows diagnostics of the fit (residuals versus predicted values, and QQ plot of residuals) for the five best models. It calls the plot functions on the fitted model objects, which should be defined (e.g. plot.lm, plot.glm). This type of plot can only be used if model objects are included in the glmulti object (i.e. if includeobjects was set to true).

type="s" plots the relative importance of model terms, i.e. the overall support for each variable across all models. A vertical line is drawn at 80

type="w" plots the normalized evidence weights of the models. A vertical line delineates models that sum um to 95

print prints a brief synthesis of the analysis (e.g. the best model found, its IC value and evidence weight...)

summary returns a list with more detailed elements:

name	the name of the analysis
method	The method used
fitting	The fitting function used
crit	The IC used
level	Whether interactions between predictors were considered or not
marginality	Whether the marginality rule was applied
confsetsize	The requested size of the confidence set
bestic	The lowest IC found
icvalues	The IC values of the models in the confidence set, from lower to greater

bestmodel	A list containing the formula of the best model found, or of the best models found if several had the same IC value
generations	The number of generations it took to converge. For genetic algorithm only.
elapsed	The actual (system) time it took. For genetic algorithm only.
includeobjects	A boolean indicating whether fitted model objects are contained in the object.

Author(s)

Vincent Calcagno, McGill University

References

Calcagno & de Mazancourt 2010 J. Stat. Soft. v34 i12. See http://www.jstatsoft.org/v34/i12

See Also

glmulti

---

weightable *Table of relative supports*

Description

Prepares a table with model formulas, IC values and IC relative supports

Usage

```
weightable(object, ...)
```

Arguments

object	A glmulti object
...	For further improvements

Value

A data frame with the list of formulas, IC values and IC weights

Author(s)

Jarret Byrnes and Vincent Calcagno, McGill University

See Also

glmulti

weightable-methods *Table of relative supports*

Description

Prepares a table with model formulas, IC values and IC relative supports

Methods

object="glmulti" Default call

write *Writing glmulti objects*

Description

The S4 method provided for `glmulti` objects writes a `glmulti` object as a data.frame, or alternatively as a raw R object.

Usage

```
write(x, file = "data",
 ncolumns = if (is.character(x)) 1 else 5,
 append = FALSE,
 sep = " "
 )
```

Arguments

`x`	An object of class glmulti
`file`	The name of the file to write into. If missing, the name is taken to be the name of the *glmulti* object.* If "\object" is at the end of the filename, then the object is written as a raw R object using *.saveRDS*.
`ncolumns`	Not used with *glmulti* objects
`append`	Whether to append the output to the file, if existing.
`sep`	The character to be used to separate columns.

Details

When using "\object", the file written can be read directly from R using *.readRDS*, or with *consensus*.

Value

A data.frame. Rows correspond to all models, sorted from best to worse.. The first columns indicate whether the different terms are found in the formula of each model (0/1). The next columns contain model complexity (K), model IC, and model formula (as a character string).

Author(s)

Vincent Calcagno, McGill University

See Also

`glmulti`, `consensus`

`write-methods` *Methods for Function write*

Description

Methods for function `write`

Methods

x = "glmulti" Method to write `glmulti` objects.

Index

∗ **IO**
write, 17
∗ **classes**
glmulti-class, 12
∗ **methods**
aic-methods, 3
aicc-methods, 3
bic-methods, 3
consensus-methods, 6
getfit-methods, 8
glmulti-methods, 13
qaic-methods, 14
qaicc-methods, 14
weightable-methods, 17
write-methods, 18
∗ **models**
aic, 2
coef.glmulti, 4
consensus, 5
glmulti, 8
summary.glmulti, 14
weightable, 16
∗ **regression**
aic, 2
coef.glmulti, 4
consensus, 5
glmulti, 8
summary.glmulti, 14
weightable, 16

aic, 2, *11*
aic,ANY-method *(*aic-methods*)*, 3
aic-methods, 3
aicc *(*aic*)*, 2
aicc,ANY-method *(*aicc-methods*)*, 3
aicc-methods, 3

bic *(*aic*)*, 2
bic,ANY-method *(*bic-methods*)*, 3
bic-methods, 3

coef.glmulti, 4, *11*, *13*
consensus, 5, *11*, *13*, *18*
consensus,list-method
*(*consensus-methods*)*, 6
consensus-methods, 6

getfit, 7
getfit,ANY-method *(*getfit-methods*)*, 8
getfit,coxph-method *(*getfit-methods*)*, 8
getfit,coxph.null-method
*(*getfit-methods*)*, 8
getfit-methods, 8
glmulti, *3*, *5–7*, 8, *16*, *18*
glmulti,ANY,ANY,ANY,ANY-method
*(*glmulti-methods*)*, 13
glmulti,character,character,ANY,ANY-method
*(*glmulti-methods*)*, 13
glmulti,character,missing,ANY,missing-method
*(*glmulti-methods*)*, 13
glmulti,formula,missing,ANY,missing-method
*(*glmulti-methods*)*, 13
glmulti,glm,missing,ANY,missing-method
*(*glmulti-methods*)*, 13
glmulti,list,ANY,ANY,ANY-method
*(*glmulti-methods*)*, 13
glmulti,lm,missing,ANY,missing-method
*(*glmulti-methods*)*, 13
glmulti,missing,ANY,ANY,ANY-method
*(*glmulti-methods*)*, 13
glmulti-class, 12
glmulti-methods, 13

plot.glmulti *(*summary.glmulti*)*, 14
predict.glmulti *(*coef.glmulti*)*, 4
print.glmulti *(*summary.glmulti*)*, 14

qaic *(*aic*)*, 2
qaic,ANY-method *(*qaic-methods*)*, 14
qaic-methods, 14
qaicc *(*aic*)*, 2

qaicc,ANY-method *(*qaicc-methods*)*, 14
qaicc-methods, 14

step, *11*
summary.glmulti, *11*, *13*, 14

weightable, *11*, 16
weightable,glmulti-method
*(*weightable-methods*)*, 17
weightable-methods, 17
write, 17
write,glmulti-method *(*write-methods*)*, 18
write-methods, 18