Package 'buildmer'

March 17, 2018

Title Stepwise Elimination and Term Reordering for Mixed-Effects Regression

Version 0.1

Description The buildmer package attempts to build, from the user's specifications, the largest possible regression model that will still converge, and then eliminates all terms from it that do not significantly contribute to an improvement of model deviance. Optional reordering of the terms by their contribution to the deviance (like SPSS) is also supported.

Depends R (>= 3.2), plyr, mgcv, lme4

Suggests lmerTest, pbkrtest, gamm4, glmmTMB

License 2-clause BSD

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

R topics documented:

add.terms *Add terms to a formula*

Description

Add terms to a formula

Usage

```
add.terms(formula, add)
```

Arguments

formula	The formula to add terms to.
add	A vector of terms to add. To add terms nested in random-effect groups, use '(term|group)' syntax if you want to add an independent random effect (e.g. '(oldertermlgroup) + (term|group)'), or use 'term|group' syntax if you want to add a dependent random effect to a pre-existing term group (if no such group exists, it will be created at the end of the formula).

Value

The updated formula.

See Also

buildmer

buildbam *Use buildmer to fit big generalized additive models using bam() from mgcv*

Description

Use buildmer to fit big generalized additive models using bam() from mgcv

Usage

```
buildbam(formula, data, family = gaussian, cl = NULL, reduce.fixed = TRUE,
  reduce.random = TRUE, direction = c("order", "backward"), crit = "LRT",
  calc.anova = TRUE, calc.summary = TRUE, ddf = "Wald", quiet = FALSE,
  ...)
```

Arguments

formula	The model formula for the maximal model you would like to fit, if possible. Supports lme4 random effects and gamm4 smooth terms.
data	The data to fit the models to.
family	The error distribution to use. Only relevant for generalized models; if the family is empty or 'gaussian', the models will be fit using lm(er), otherwise they will be fit using glm(er) with the specified error distribution passed through.
cl	An optional cluster object as returned by parallel::makeCluster() to use for parallelizing the evaluation of terms.
reduce.fixed	Whether to reduce the fixed-effect structure.
reduce.random	Whether to reduce the random-effect structure.
direction	The direction for stepwise elimination; possible options are 'order' (order terms by their contribution to the model), 'backward' (backward elimination), 'forward' (forward elimination, implies 'order'). The default is the combination 'c('order','backward')', to first make sure that the model converges and to then perform backward elimination; other such combinations are perfectly allowed.
crit	The criterion used to test terms for elimination. Possible options are 'LRT' (default), 'AIC', and 'BIC'.
calc.anova	Whether to also calculate the ANOVA table for the final model after term elimination.
calc.summary	Whether to also calculate the summary table for the final model after term elimination.
quiet	Whether to suppress progress messages.
...	Additional options to be passed to bam().

Value

A buildmer object containing the following slots:

- model: the final model containing only the terms that survived elimination
- p: the parameter list used in the various buildmer modules. Things of interest this list includes are, among others:
  - results: a dataframe containing the results of the elimination process
  - messages: any warning messages

  This information is also printed as part of the show() method.
- summary: the model's summary, if 'calc.summary=TRUE' was passed
- anova: the model's anova, if 'calc.anova=TRUE' was passed

See Also

buildmer

`buildgam` *Use buildmer to fit generalized additive models using gam() from mgcv*

Description

Use buildmer to fit generalized additive models using gam() from mgcv

Usage

```
buildgam(formula, data, family = gaussian, cl = NULL, reduce.fixed = TRUE,
  reduce.random = TRUE, direction = c("order", "backward"), crit = "LRT",
  calc.anova = TRUE, calc.summary = TRUE, ddf = "Wald", quiet = FALSE,
  ...)
```

Arguments

`formula`	The model formula for the maximal model you would like to fit, if possible.
`data`	The data to fit the models to.
`family`	The error distribution to use. Only relevant for generalized models; if the family is empty or 'gaussian', the models will be fit using lm(er), otherwise they will be fit using glm(er) with the specified error distribution passed through.
`cl`	An optional cluster object as returned by parallel::makeCluster() to use for parallelizing the evaluation of terms.
`reduce.fixed`	Whether to reduce the fixed-effect structure.
`reduce.random`	Whether to reduce the random-effect structure.
`direction`	The direction for stepwise elimination; possible options are 'order' (order terms by their contribution to the model), 'backward' (backward elimination), 'forward' (forward elimination, implies 'order'). The default is the combination 'c('order','backward')', to first make sure that the model converges and to then perform backward elimination; other such combinations are perfectly allowed.
`crit`	The criterion used to test terms for elimination. Possible options are 'LRT' (default), 'AIC', and 'BIC'.
`calc.anova`	Whether to also calculate the ANOVA table for the final model after term elimination.
`calc.summary`	Whether to also calculate the summary table for the final model after term elimination.
`quiet`	Whether to suppress progress messages.
`...`	Additional options to be passed to gam().

Value

A buildmer object containing the following slots:

- model: the final model containing only the terms that survived elimination
- p: the parameter list used in the various buildmer modules. Things of interest this list includes are, among others:
  - results: a dataframe containing the results of the elimination process

  - messages: any warning messages

  This information is also printed as part of the show() method.

- summary: the model's summary, if 'calc.summary=TRUE' was passed
- anova: the model's anova, if 'calc.anova=TRUE' was passed

See Also

buildmer

`buildgamm` *The logical extension of buildgam() to buildgamm() is not supported, because (i) gamm assumes you know what you're doing; (ii) the log-likelihood of a gamm object's 'lme' item is not actually the log-likelihood of the final model; (iii) in my experience, gamm fits often fail to converge. If you are only using gamm for its 'true' random effects, use buildgamm4(). If you are using gamm for correlation structures, use buildglmmTMB(), or buildbam() if AR(1) will do and your errors are normal. If you want more complex correlation structures, perform the stepwise elimination process by hand...*

Description

The logical extension of buildgam() to buildgamm() is not supported, because (i) gamm assumes you know what you're doing; (ii) the log-likelihood of a gamm object's 'lme' item is not actually the log-likelihood of the final model; (iii) in my experience, gamm fits often fail to converge. If you are only using gamm for its 'true' random effects, use buildgamm4(). If you are using gamm for correlation structures, use buildglmmTMB(), or buildbam() if AR(1) will do and your errors are normal. If you want more complex correlation structures, perform the stepwise elimination process by hand...

Usage

```
buildgamm(...)
```

See Also

buildgamm4, buildbam, buildgam

`buildgamm4` *Use buildmer to fit generalized additive models using gamm4*

Description

Use buildmer to fit generalized additive models using gamm4

Usage

```
buildgamm4(...)
```

Arguments

...	Additional options to be passed to gam().
formula	The model formula for the maximal model you would like to fit, if possible. Supports lme4 random effects and gamm4 smooth terms.
data	The data to fit the models to.
family	The error distribution to use. Only relevant for generalized models; if the family is empty or 'gaussian', the models will be fit using lm(er), otherwise they will be fit using glm(er) with the specified error distribution passed through.
cl	An optional cluster object as returned by parallel::makeCluster() to use for parallelizing the evaluation of terms.
reduce.fixed	Whether to reduce the fixed-effect structure.
reduce.random	Whether to reduce the random-effect structure.
direction	The direction for stepwise elimination; possible options are 'order' (order terms by their contribution to the model), 'backward' (backward elimination), 'forward' (forward elimination, implies 'order'). The default is the combination 'c('order','backward')', to first make sure that the model converges and to then perform backward elimination; other such combinations are perfectly allowed.
crit	The criterion used to test terms for elimination. Possible options are 'LRT' (default), 'AIC', and 'BIC'.
calc.anova	Whether to also calculate the ANOVA table for the final model after term elimination.
calc.summary	Whether to also calculate the summary table for the final model after term elimination.
ddf	The method used for calculating p-values if all smooth terms were eliminated and summary=TRUE. Options are 'Wald' (default), 'Satterthwaite' (if lmerTest is available), 'Kenward-Roger' (if lmerTest and pbkrtest are available), and 'lme4' (no p-values).
quiet	Whether to suppress progress messages.

Value

A buildmer object containing the following slots:

- model: the final model containing only the terms that survived elimination
- p: the parameter list used in the various buildmer modules. Things of interest this list includes are, among others:
  - results: a dataframe containing the results of the elimination process
  - messages: any warning messages

  This information is also printed as part of the show() method.
- summary: the model's summary, if 'calc.summary=TRUE' was passed
- anova: the model's anova, if 'calc.anova=TRUE' was passed

See Also

buildmer

buildglmmTMB *Use buildmer to perform stepwise elimination on glmmTMB models*

Description

Use buildmer to perform stepwise elimination on glmmTMB models

Usage

```
buildglmmTMB(formula, data, family = gaussian, correlation = NULL,
  cl = NULL, reduce.fixed = TRUE, reduce.random = TRUE,
  direction = c("order", "backward"), crit = "LRT", calc.anova = TRUE,
  calc.summary = TRUE, ddf = "Wald", quiet = FALSE, ...)
```

Arguments

formula	The model formula for the maximal model you would like to fit, if possible.
data	The data to fit the models to.
family	The error distribution to use. Only relevant for generalized models.
correlation	Contrary to normal glmmTMB usage, correlation structures such as 'ar1(0+covariate|grouping)' need to be specified in a separate argument in plain text to prevent them from being eliminated (and to work around a problem in lme4:::findbars()). The correct usage is 'buildglmmTMB(formula,data,family,correlation="ar1(0+covariate|grouping)")'.
cl	An optional cluster object as returned by parallel::makeCluster() to use for parallelizing the evaluation of terms.
reduce.fixed	Whether to reduce the fixed-effect structure.
reduce.random	Whether to reduce the random-effect structure.
direction	The direction for stepwise elimination; possible options are 'order' (order terms by their contribution to the model), 'backward' (backward elimination), 'forward' (forward elimination, implies 'order'). The default is the combination 'c('order','backward')', to first make sure that the model converges and to then perform backward elimination; other such combinations are perfectly allowed.
crit	The criterion used to test terms for elimination. Possible options are 'LRT', 'AIC', and 'BIC'.
calc.anova	Whether to also calculate the ANOVA table for the final model after term elimination.
calc.summary	Whether to also calculate the summary table for the final model after term elimination.
quiet	Whether to suppress progress messages.
...	Additional options to be passed to glmmTMB().

Value

A buildmer object containing the following slots:

- model: the final model containing only the terms that survived elimination
- p: the parameter list used in the various buildmer modules. Things of interest this list includes are, among others:

  - results: a dataframe containing the results of the elimination process
  - messages: any warning messages

  This information is also printed as part of the show() method.

- summary: the model's summary, if 'calc.summary=TRUE' was passed
- anova: the model's anova, if 'calc.anova=TRUE' was passed

See Also

buildmer

`buildgls` *Use buildmer to fit generalized-least-squares models using gls() from nlme*

Description

Use buildmer to fit generalized-least-squares models using gls() from nlme

Usage

```
buildgls(formula, data, random, cl = NULL, reduce.fixed = TRUE,
  direction = c("order", "backward"), crit = "LRT", calc.anova = TRUE,
  calc.summary = TRUE, quiet = FALSE, ...)
```

Arguments

`formula`	The model formula for the maximal model you would like to fit, if possible.
`data`	The data to fit the models to.
`cl`	An optional cluster object as returned by parallel::makeCluster() to use for parallelizing the evaluation of terms.
`reduce.fixed`	Whether to reduce the fixed-effect structure.
`direction`	The direction for stepwise elimination; possible options are 'order' (order terms by their contribution to the model), 'backward' (backward elimination), 'forward' (forward elimination, implies 'order'). The default is the combination 'c('order','backward')', to first make sure that the model converges and to then perform backward elimination; other such combinations are perfectly allowed.
`crit`	The criterion used to test terms for elimination. Possible options are 'LRT' (default), 'AIC', and 'BIC'.
`calc.anova`	Whether to also calculate the ANOVA table for the final model after term elimination.
`calc.summary`	Whether to also calculate the summary table for the final model after term elimination.
`quiet`	Whether to suppress progress messages.
`...`	Additional options to be passed to lme().

Value

A buildmer object containing the following slots:

- model: the final model containing only the terms that survived elimination
- p: the parameter list used in the various buildmer modules. Things of interest this list includes are, among others:
  - results: a dataframe containing the results of the elimination process
  - messages: any warning messages

  This information is also printed as part of the show() method.
- summary: the model's summary, if 'calc.summary=TRUE' was passed
- anova: the model's anova, if 'calc.anova=TRUE' was passed

See Also

buildmer

`buildlme` *Use buildmer to perform stepwise elimination of the fixed-effects part of mixed-effects models fit via lme() from nlme*

Description

Use buildmer to perform stepwise elimination of the fixed-effects part of mixed-effects models fit via lme() from nlme

Usage

```
buildlme(formula, data, random, cl = NULL, reduce.fixed = TRUE,
  direction = c("order", "backward"), crit = "LRT", calc.anova = TRUE,
  calc.summary = TRUE, quiet = FALSE, ...)
```

Arguments

`formula`	The model formula for the maximal model you would like to fit, if possible.
`data`	The data to fit the models to.
`random`	The random-effects specification for the model. This is not manipulated by buildlme() in any way!
`cl`	An optional cluster object as returned by parallel::makeCluster() to use for parallelizing the evaluation of terms.
`reduce.fixed`	Whether to reduce the fixed-effect structure.
`direction`	The direction for stepwise elimination; possible options are 'order' (order terms by their contribution to the model), 'backward' (backward elimination), 'forward' (forward elimination, implies 'order'). The default is the combination 'c('order','backward')', to first make sure that the model converges and to then perform backward elimination; other such combinations are perfectly allowed.
`crit`	The criterion used to test terms for elimination. Possible options are 'LRT', 'AIC', and 'BIC'.

calc.anova	Whether to also calculate the ANOVA table for the final model after term elimination.
calc.summary	Whether to also calculate the summary table for the final model after term elimination.
quiet	Whether to suppress progress messages.
...	Additional options to be passed to lme().

Value

A buildmer object containing the following slots:

- model: the final model containing only the terms that survived elimination
- p: the parameter list used in the various buildmer modules. Things of interest this list includes are, among others:
  - results: a dataframe containing the results of the elimination process
  - messages: any warning messages

  This information is also printed as part of the show() method.
- summary: the model's summary, if 'calc.summary=TRUE' was passed
- anova: the model's anova, if 'calc.anova=TRUE' was passed

See Also

buildmer

---

buildmer *Construct and fit as complete a model as possible, optionally order terms by their contribution to the log-likelihood, and perform stepwise elimination using the change in log-likelihood*

Description

Construct and fit as complete a model as possible, optionally order terms by their contribution to the log-likelihood, and perform stepwise elimination using the change in log-likelihood

Usage

```
buildmer(formula, data, family = gaussian, cl = NULL, reduce.fixed = TRUE,
  reduce.random = TRUE, direction = c("order", "backward"), crit = "LRT",
  calc.anova = TRUE, calc.summary = TRUE, ddf = "Wald", quiet = FALSE,
  ...)
```

Arguments

formula	The model formula for the maximal model you would like to fit, if possible. Supports lme4 random effects and gamm4 smooth terms.
data	The data to fit the models to.
family	The error distribution to use. Only relevant for generalized models; if the family is empty or 'gaussian', the models will be fit using lm(er), otherwise they will be fit using glm(er) with the specified error distribution passed through.

`cl`	An optional cluster object as returned by parallel::makeCluster() to use for parallelizing the evaluation of terms.
`reduce.fixed`	Whether to reduce the fixed-effect structure.
`reduce.random`	Whether to reduce the random-effect structure.
`direction`	The direction for stepwise elimination; possible options are 'order' (order terms by their contribution to the model), 'backward' (backward elimination), 'forward' (forward elimination, implies 'order'). The default is the combination 'c('order','backward')', to first make sure that the model converges and to then perform backward elimination; other such combinations are perfectly allowed.
`crit`	The criterion used to test terms for elimination. Possible options are 'LRT' (default), 'AIC', and 'BIC'.
`calc.anova`	Whether to also calculate the ANOVA table for the final model after term elimination. This is useful if you want to calculate degrees of freedom by Kenward-Roger approximation, in which case generating the ANOVA table (via lmerTest) will be very slow, and preparing the ANOVA in advance can be advantageous.
`calc.summary`	Whether to also calculate the summary table for the final model after term elimination. This is useful if you want to calculate degrees of freedom by Kenward-Roger approximation (default), in which case generating the summary (via lmerTest) will be very slow, and preparing the summary in advance can be advantageous.
`ddf`	The method used for calculating p-values if summary=TRUE. Options are 'Wald' (default), 'Satterthwaite' (if lmerTest is available), 'Kenward-Roger' (if lmerTest and pbkrtest are available), and 'lme4' (no p-values).
`quiet`	Whether to suppress progress messages.
`...`	Additional options to be passed to (g)lmer or gamm4. (They will also be passed to (g)lm in so far as they're applicable, so you can use arguments like 'subset=...' and expect things to work. The single exception is the 'control' argument, which is assumed to be meant only for (g)lmer and not for (g)lm, and will NOT be passed on to (g)lm.)

Value

A buildmer object containing the following slots:

- model: the final model containing only the terms that survived elimination
- p: the parameter list used in the various buildmer modules. Things of interest this list includes are, among others:
  - results: a dataframe containing the results of the elimination process
  - messages: any warning messages

  This information is also printed as part of the show() method.
- summary: the model's summary, if 'calc.summary=TRUE' was passed
- anova: the model's anova, if 'calc.anova=TRUE' was passed

Examples

```
buildmer(Reaction~Days+(Days|Subject),sleepstudy)
```

buildmer-class *The buildmer class*

Description

The buildmer class

Arguments

model	The final model containing only the terms that survived elimination.
p	Parameters used during the fitting process.
anova	The model's ANOVA, if the model was built with 'anova=TRUE'.
summary	The model's summary, if the model was built with 'summary=TRUE'.

See Also

buildmer

calcWald *Calculate p-values based on Wald z-scores*

Description

Calculate p-values based on Wald z-scores

Usage

```
calcWald(table, i, sqrt = FALSE)
```

Arguments

table	A coefficient table from a summary or anova output.
i	The number of the column in that table containing the t-values.
sqrt	Whether we're testing F values or t values (default).

Value

The table augmented with p-values.

conv — *Test a mixed model for convergence*

Description

Test a mixed model for convergence

Usage

```
conv(model)
```

Arguments

model	The model object to test.

Value

Whether the model converged or not.

diag,formula-method — *Diagonalize the random-effect covariance structure, possibly assisting convergence*

Description

Diagonalize the random-effect covariance structure, possibly assisting convergence

Usage

```
## S4 method for signature 'formula'
diag(x)
```

Arguments

formula	A model formula.

Value

The formula with all random-effect correlations forced to zero, per Pinheiro & Bates (2000).

has.smooth.terms *Test whether a formula contains mgcv smooth terms*

Description

Test whether a formula contains mgcv smooth terms

Usage

```
has.smooth.terms(formula)
```

Arguments

formula	The formula.

Value

A logical indicating whether the formula has any gamm4 terms.

hasREML *Test whether a model was fit with REML*

Description

Test whether a model was fit with REML

Usage

```
hasREML(model)
```

Arguments

model	A fitted model object.

Value

TRUE or FALSE if the model was a linear mixed-effects model that was fit with REML or not, respectively; NA otherwise.

is.random.term *Test whether a formula term contains lme4 random terms*

Description

Test whether a formula term contains lme4 random terms

Usage

```
is.random.term(term)
```

Arguments

term	The term.

Value

A logical indicating whether the term was a random-effects term.

is.smooth.term *Test whether a formula term is an mgcv smooth term*

Description

Test whether a formula term is an mgcv smooth term

Usage

```
is.smooth.term(term)
```

Arguments

term	The term.

Value

A logical indicating whether the term was a random-effects term.

remove.terms *Remove terms from an lme4 formula*

Description

Remove terms from an lme4 formula

Usage

```
remove.terms(formula, remove)
```

Arguments

formula	The lme4 formula.
remove	A vector of terms to remove. To remove terms nested inside random-effect groups, use 'term|group' syntax. Note that marginality is respected, i.e. no effects will be removed if they participate in a higher-order interaction, and no fixed effects will be removed if a random slope is included over that fixed effect.
formulize	Whether to return a formula (default) or a simple list of terms.

See Also

buildmer

stepwise *A simple interface to buildmer intended to mimic SPSS stepwise methods for term ordering and backward stepwise elimination*

Description

A simple interface to buildmer intended to mimic SPSS stepwise methods for term ordering and backward stepwise elimination

Usage

```
stepwise(formula, data, family = gaussian, ...)
```

Arguments

formula	The model formula for the maximal model you would like to fit, if possible. Supports lme4 random effects and gamm4 smooth terms.
data	The data to fit the models to.
family	The error distribution to use. Only relevant for generalized models; if the family is empty or 'gaussian', the models will be fit using lm(er), otherwise they will be fit using glm(er) with the specified error distribution passed through. Commonly-used options are either nothing/'gaussian' (linear regression), 'binomial' (logistic regression), or 'poisson' (loglin regression), although many other families exist (e.g. cloglog, ...).
...	Additional parameters that override buildmer defaults, see 'buildmer'.

Value

A buildmer object, which you can use summary() on to get a summary of the final model.

Examples

```
stepwise(Reaction~Days+(Days|Subject),sleepstudy)
```

---

vowels *Vowel data from a pilot study.*

Description

Vowel data from a pilot study.

Usage

```
data(vowels)
```

Format

A standard data frame.

Examples

```
#buildmer(f1 ~ vowel*timepoint*following + stress + information + (vowel*timepoint*following|participant),dat
buildmer(f1 ~ vowel + timepoint + stress + following + information + vowel:timepoint + timepoint:following + vow
```

Index