Package 'buildmer'

May 20, 2019

Title Stepwise Elimination and Term Reordering for Mixed-Effects Regression

Version 1.1

Description Finds the largest possible regression model that will still converge for various types of regression analyses (including mixed models and generalized additive models) and then optionally performs stepwise elimination similar to the forward and backward effect selection methods in SAS, based on the change in log-likelihood, Akaike's Information Criterion, or the Bayesian Information Criterion.

Depends R (>= 3.2)

Imports methods, mgcv, lme4, plyr, stats, utils

Suggests JuliaCall, MASS, gamm4, glmmTMB, knitr, lmerTest, nlme, nnet, parallel, pbkrtest, rmarkdown

License FreeBSD

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

BugReports https://github.com/cvoeten/buildmer/issues

VignetteBuilder knitr

NeedsCompilation no

Author Cesko C. Voeten [aut, cre]

Maintainer Cesko C. Voeten <cvoeten@gmail.com>

Repository CRAN

Date/Publication 2019-05-19 22:30:12 UTC

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. '(olderterm|group) + (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.

Examples

```
library(buildmer)
form <- Reaction ~ Days + (1|Subject)
add.terms(form,'Days|Subject')
add.terms(form,'(0+Days|Subject)')
```

build.formula *Convert a buildmer term list into a proper model formula*

Description

Convert a buildmer term list into a proper model formula

Usage

```
build.formula(dep, terms)
```

Arguments

dep	The dependent variable.
terms	The term list.

Value

A formula.

Examples

```
library(buildmer)
form1 <- Reaction ~ Days + (Days|Subject)
terms <- tabulate.formula(form1)
form2 <- build.formula(dep='Reaction',terms)

# check that the two formulas give the same results
library(lme4)
check <- function (f) resid(lmer(f,sleepstudy))
all.equal(check(form1),check(form2))
```

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

Description

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

Usage

```
buildbam(formula, data = NULL, family = "gaussian", cl = NULL,
  direction = c("order", "backward"), crit = "LRT", include = NULL,
  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.
family	The error distribution to use.
cl	An optional cluster object as returned by function makeCluster() from package parallel to use for parallelizing the evaluation of terms.
direction	Character string or vector indicating 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	Character string or vector determining the criterion used to test terms for elimination. Possible options are 'LRT' (likelihood-ratio test; this is the default), 'LL' (use the raw -2 log likelihood), 'AIC' (Akaike Information Criterion), and 'BIC' (Bayesian Information Criterion).
include	A character vector of terms that will be kept in the model at all times. These do not need to be specified separately in the formula argument.
calc.anova	Logical indicating whether to also calculate the ANOVA table for the final model after term elimination.
calc.summary	Logical indicating whether to also calculate the summary table for the final model after term elimination.
quiet	Logical indicating whether to suppress progress messages.
...	Additional options to be passed to bam().

See Also

buildmer()

Examples

```
library(buildmer)
m <- buildbam(f1 ~ s(timepoint,by=following) + s(participant,by='following',bs='re') +
                   s(participant,timepoint,by=following,bs='fs'),data=vowels)
```

`buildcustom` *Use buildmer to perform stepwise elimination using a custom fitting function*

Description

Use buildmer to perform stepwise elimination using a custom fitting function

Usage

```
buildcustom(formula, cl = NULL, direction = c("order", "backward"),
  crit = function(ref, alt) stop("'crit' not specified"),
  include = NULL, reduce.fixed = T, reduce.random = T,
  fit = function(p, formula) stop("'fit' not specified"),
  elim = function(x) stop("'elim' not specified"), quiet = FALSE, ...)
```

Arguments

`formula`	The model formula for the maximal model you would like to fit, if possible.
`cl`	An optional cluster object as returned by function `makeCluster()` from package `parallel` to use for parallelizing the evaluation of terms.
`direction`	Character string or vector indicating 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`	A function taking two arguments and outputting a single score, denoting the difference between the models. This can also be a character string or vector of any of `'LRT'` (likelihood-ratio test), `'LL'` (use the raw -2 log likelihood), `'AIC'` (Akaike Information Criterion), and `'BIC'` (Bayesian Information Criterion).
`include`	A one-sided formula whose terms will always be included in the model formula. Useful for e.g.\ passing correlation structures in `glmmTMB` models.
`reduce.fixed`	Logical indicating whether to reduce the fixed-effect structure.
`reduce.random`	Logical indicating whether to reduce the random-effect structure.
`fit`	A function taking two arguments, of which the first is the `buildmer` parameter list p and the second one is a formula. The function must return a single object, which is treated as a model object fitted via the provided formula. The function must return an error (‘`stop()`’) if the model does not converge.
`elim`	A function taking one argument and returning a single value. The first argument is the return value of the function passed in `crit`, and the returned value must be a logical indicating if the small model must be selected (return `TRUE`) or the large model (return `FALSE`).
`quiet`	Logical indicating whether to suppress progress messages.
`...`	Additional options to be passed to the fitting function, such as perhaps a `data` argument.

See Also

buildmer()

Examples

```
## Use buildmer to do stepwise linear discriminant analysis
library(buildmer)
migrant[,-1] <- scale(migrant[,-1])
flipfit <- function (p,formula) {
    # The predictors must be entered as dependent variables in a MANOVA
    # (i.e. the predictors must be flipped with the dependent variable)
    Y <- model.matrix(formula,migrant)
    m <- lm(Y ~ 0+migrant$changed)
    # the model may error out when asking for the MANOVA
    test <- try(anova(m))
    if (inherits(test,'try-error')) test else m
}
crit.F <- function (ma,mb) { # use whole-model F
    pvals <- anova(mb)$'Pr(>F)' # not valid for backward!
    pvals[length(pvals)-1]
}
crit.Wilks <- function (ma,mb) {
   if (is.null(ma)) return(crit.F(ma,mb)) #not completely correct, but close as F approximates X2
    Lambda <- anova(mb,test='Wilks')$Wilks[1]
    p <- length(coef(mb))
    n <- 1
    m <- nrow(migrant)
    Bartlett <- ((p-n+1)/2-m)*log(Lambda)
    pchisq(Bartlett,n*p,lower.tail=FALSE)
}

# First, order the terms based on Wilks' Lambda
m <- buildcustom(changed ~ friends.nl+friends.be+multilingual+standard+hearing+reading+attention+
sleep+gender+handedness+diglossic+age+years,direction='order',fit=flipfit,crit=crit.Wilks)
# Now, use the six most important terms (arbitrary choice) in the LDA
library(MASS)
m <- lda(changed ~ diglossic + age + reading + friends.be + years + multilingual,data=migrant)
```

---

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

---

Description

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

Usage

```
buildgam(formula, data = NULL, family = "gaussian", cl = NULL,
  direction = c("order", "backward"), crit = "LRT", include = NULL,
  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.
family	The error distribution to use.
cl	An optional cluster object as returned by function makeCluster() from package parallel to use for parallelizing the evaluation of terms.
direction	Character string or vector indicating 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	Character string or vector determining the criterion used to test terms for elimination. Possible options are 'LRT' (likelihood-ratio test; this is the default), 'LL' (use the raw -2 log likelihood), 'AIC' (Akaike Information Criterion), and 'BIC' (Bayesian Information Criterion).
include	A character vector of terms that will be kept in the model at all times. These do not need to be specified separately in the formula argument.
calc.anova	Logical indicating whether to also calculate the ANOVA table for the final model after term elimination.
calc.summary	Logical indicating whether to also calculate the summary table for the final model after term elimination.
quiet	Logical indicating whether to suppress progress messages.
...	Additional options to be passed to bam().

See Also

buildmer()

Examples

```
library(buildmer)
m <- buildgam(f1 ~ s(timepoint,by=following) + s(participant,by='following',bs='re') +
                   s(participant,timepoint,by=following,bs='fs'),data=vowels)
```

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

Description

Use buildmer to fit generalized additive models using package gamm4

Usage

```
buildgamm4(formula, data = NULL, family = "gaussian", cl = NULL,
  direction = c("order", "backward"), crit = "LRT", include = NULL,
  reduce.fixed = TRUE, reduce.random = TRUE, 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.
cl	An optional cluster object as returned by function `makeCluster()` from package `parallel` to use for parallelizing the evaluation of terms.
direction	Character string or vector indicating 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	Character string or vector determining the criterion used to test terms for elimination. Possible options are `'LRT'` (likelihood-ratio test; this is the default), `'LL'` (use the raw -2 log likelihood), `'AIC'` (Akaike Information Criterion), and `'BIC'` (Bayesian Information Criterion).
include	A character vector of terms that will be kept in the model at all times. These do not need to be specified separately in the `formula` argument.
reduce.fixed	Logical indicating whether to reduce the fixed-effect structure.
reduce.random	Logical indicating whether to reduce the random-effect structure.
calc.anova	Logical indicating whether to also calculate the ANOVA table for the final model after term elimination.
calc.summary	Logical indicating 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 `calc.summary=TRUE`. Options are `'Wald'` (default), `'Satterthwaite'` (if package `lmerTest` is available), `'Kenward-Roger'` (if packages `lmerTest` and `pbkrtest` are available), and `'lme4'` (no *p*-values).
quiet	Logical indicating whether to suppress progress messages.
...	Additional options to be passed to `gamm4()`.

See Also

buildmer()

Examples

```
library(buildmer)
m <- buildgamm4(f1 ~ s(timepoint,by=following) +
                     s(participant,timepoint,by=following,bs='fs'),data=vowels)
```

---

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

Description

Use buildmer to perform stepwise elimination on glmmTMB models

Usage

```
buildglmmTMB(formula, data = NULL, family = "gaussian", cl = NULL,
  direction = c("order", "backward"), crit = "LRT", include = NULL,
  reduce.fixed = TRUE, reduce.random = 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.
family	The error distribution to use.
cl	An optional cluster object as returned by function makeCluster() from package parallel to use for parallelizing the evaluation of terms.
direction	Character string or vector indicating 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	Character string or vector determining the criterion used to test terms for elimination. Possible options are 'LRT' (likelihood-ratio test; this is the default), 'LL' (use the raw -2 log likelihood), 'AIC' (Akaike Information Criterion), and 'BIC' (Bayesian Information Criterion).
include	A character vector of terms that will be kept in the model at all times. These do not need to be specified separately in the formula argument.

reduce.fixed	Logical indicating whether to reduce the fixed-effect structure.
reduce.random	Logical indicating whether to reduce the random-effect structure.
calc.summary	Logical indicating whether to also calculate the summary table for the final model after term elimination.
quiet	Logical indicating whether to suppress progress messages.
...	Additional options to be passed to glmmTMB().

See Also

buildmer()

Examples

```
library(buildmer)
m <- buildglmmTMB(Reaction ~ Days + (Days|Subject),lme4::sleepstudy)
```

---

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 = NULL, cl = NULL, direction = c("order",
  "backward"), crit = "LRT", include = NULL, 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 function makeCluster() from package parallel to use for parallelizing the evaluation of terms.
direction	Character string or vector indicating 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	Character string or vector determining the criterion used to test terms for elimination. Possible options are 'LRT' (likelihood-ratio test; this is the default), 'LL' (use the raw -2 log likelihood), 'AIC' (Akaike Information Criterion), and 'BIC' (Bayesian Information Criterion).
include	A character vector of terms that will be kept in the model at all times. These do not need to be specified separately in the formula argument.
calc.anova	Logical indicating whether to also calculate the ANOVA table for the final model after term elimination.
calc.summary	Logical indicating whether to also calculate the summary table for the final model after term elimination.
quiet	Logical indicating whether to suppress progress messages.
...	Additional options to be passed to gls().

See Also

buildmer()

Examples

```
library(buildmer)
library(nlme)
vowels$event <- with(vowels,interaction(participant,word))
m <- buildgls(f1 ~ timepoint*following,correlation=corAR1(form=~1|event),data=vowels)
```

---

buildjulia *Use buildmer to perform stepwise elimination on models fit with Julia package MixedModels via JuliaCall*

Description

Use buildmer to perform stepwise elimination on models fit with Julia package MixedModels via JuliaCall

Usage

```
buildjulia(formula, data = NULL, family = "gaussian", include = NULL,
  julia_family = NULL, julia_link = NULL, julia_fun = NULL,
  direction = c("order", "backward"), crit = "LRT",
  reduce.fixed = TRUE, reduce.random = 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.
family	The error distribution to use.

include	A character vector of terms that will be kept in the model at all times. These do not need to be specified separately in the formula argument.
julia_family	For generalized linear mixed models, the name of the Julia function to evaluate to obtain the error distribution. Only used if family is empty or gaussian. This should probably be the same as family but with an initial capital, with the notable exception of logistic regression: if the R family is binomial, the Julia family should be 'Bernoulli'.
julia_link	For generalized linear mixed models, the name of the Julia function to evaluate to obtain the link function. Only used if family is empty or gaussian. If not provided, Julia's default link for your error distribution is used.
julia_fun	If you need to change some parameters in the Julia model object before Julia fit! is called, you can provide an R function to manipulate the unfitted Julia object here. This function should accept two arguments: the first is the julia structure, which is a list containing a call element you can use as a function to call Julia; the second argument is the R JuliaObject corresponding to the unfitted Julia model. This can be used to e.g. change optimizer parameters before the model is fitted.
direction	Character string or vector indicating 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	Character string or vector determining the criterion used to test terms for elimination. Possible options are 'LRT' (likelihood-ratio test; this is the default), 'LL' (use the raw -2 log likelihood), 'AIC' (Akaike Information Criterion), and 'BIC' (Bayesian Information Criterion).
reduce.fixed	Logical indicating whether to reduce the fixed-effect structure.
reduce.random	Logical indicating whether to reduce the random-effect structure.
quiet	Logical indicating whether to suppress progress messages.
...	Additional options to be passed to LinearMixedModel() or GeneralizedLinearMixedModel().

See Also

buildmer()

Examples

```
library(buildmer)
m <- buildjulia(f1 ~ vowel*timepoint*following + (vowel*timepoint*following|participant) +
                (timepoint|word),data=vowels)
```

`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 = NULL, random, cl = NULL,
  direction = c("order", "backward"), crit = "LRT", include = NULL,
  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 function `makeCluster()` from package `parallel` to use for parallelizing the evaluation of terms.
`direction`	Character string or vector indicating 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`	Character string or vector determining the criterion used to test terms for elimination. Possible options are `'LRT'` (likelihood-ratio test; this is the default), `'LL'` (use the raw -2 log likelihood), `'AIC'` (Akaike Information Criterion), and `'BIC'` (Bayesian Information Criterion).
`include`	A character vector of terms that will be kept in the model at all times. These do not need to be specified separately in the `formula` argument.
`calc.anova`	Logical indicating whether to also calculate the ANOVA table for the final model after term elimination.
`calc.summary`	Logical indicating whether to also calculate the summary table for the final model after term elimination.
`quiet`	Logical indicating whether to suppress progress messages.
`...`	Additional options to be passed to `lme()`.

See Also

`buildmer()`

Examples

```
library(buildmer)
m <- buildlme(Reaction ~ Days,data=lme4::sleepstudy,random=~Days|Subject)
```

`buildmer` *Construct and fit as complete a model as possible and perform stepwise elimination*

Description

With the default options, buildmer() will do two things:

1. Determine the order of the effects in your model, based on their contribution to the log-likelihood. This identifies the 'maximal model', which is the model containing either all effects specified by the user, or subset of those effects that still allow the model to converge, ordered such that the most information-rich effects have made it in.
2. Perform backward stepwise elimination based on the change in log-likelihood.

The final model is returned in the `model` slot of the returned `buildmer` object. All functions in the `buildmer` package are aware of the distinction between (f)REML and ML, and know to divide chi-square *p*-values by 2 when comparing models differing only in random slopes (see Pinheiro & Bates 2000). The steps executed above can be changed using the `direction` argument, allowing for arbitrary chains of, for instance, forward-backward-forward stepwise elimination (although using more than one elimination method on the same data is not recommended). The criterion for determining the importance of terms in the ordering stage and the elimination of terms in the elimination stage can also be changed, using the *crit* argument.

Usage

```
buildmer(formula, data = NULL, family = "gaussian", cl = NULL,
  direction = c("order", "backward"), crit = "LRT", include = NULL,
  reduce.fixed = TRUE, reduce.random = TRUE, 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.
`cl`	An optional cluster object as returned by function `makeCluster()` from package `parallel` to use for parallelizing the evaluation of terms.
`direction`	Character string or vector indicating 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	Character string or vector determining the criterion used to test terms for elimination. Possible options are 'LRT' (likelihood-ratio test; this is the default), 'LL' (use the raw -2 log likelihood), 'AIC' (Akaike Information Criterion), and 'BIC' (Bayesian Information Criterion).
include	A character vector of terms that will be kept in the model at all times. These do not need to be specified separately in the formula argument.
reduce.fixed	Logical indicating whether to reduce the fixed-effect structure.
reduce.random	Logical indicating whether to reduce the random-effect structure.
calc.anova	Logical indicating whether to also calculate the ANOVA table for the final model after term elimination.
calc.summary	Logical indicating whether to also calculate the summary table for the final model after term elimination.
ddf	The method used for calculating *p*-values if calc.anova=TRUE or calc.summary=TRUE. Options are 'Wald' (default), 'Satterthwaite' (if package lmerTest is available), 'Kenward-Roger' (if packages lmerTest and pbkrtest are available), and 'lme4' (no *p*-values).
quiet	Logical indicating whether to suppress progress messages.
...	Additional options to be passed to lme4 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 lme4 and not for (g)lm, and will *not* be passed on to (g)lm.)

Examples

```
library(buildmer)
m <- buildmer(Reaction ~ Days + (Days|Subject),lme4::sleepstudy)


# Only finding the maximal model, with importance of effects measured by AIC, parallelizing the
# model evaluations using two cores, using the bobyqa optimizer and asking for verbose output
library(parallel)
cl <- makeCluster(2,outfile='')
control <- lme4::lmerControl(optimizer='bobyqa')
clusterExport(cl,'control') #this is not done automatically for '...' arguments!
m <- buildmer(f1 ~ vowel*timepoint*following + (vowel*timepoint*following|participant) +
           (timepoint|word),data=vowels,cl=cl,direction='order',crit='AIC',calc.anova=FALSE,
              calc.summary=FALSE,control=control,verbose=2)
# The maximal model is: f1 ~ vowel + timepoint + vowel:timepoint + following +
# timepoint:following +vowel:following + vowel:timepoint:following + (1 + timepoint +
# following + timepoint:following | participant) + (1 + timepoint | word)
# Now do backward stepwise elimination (result: f1 ~ vowel + timepoint + vowel:timepoint +
# following + timepoint:following + (1 + timepoint + following + timepoint:following |
# participant) + (1 + timepoint | word))
buildmer(formula(m@model),data=vowels,direction='backward',crit='AIC',control=control)
# Or forward (result: retains the full model)
buildmer(formula(m@model),data=vowels,direction='forward',crit='AIC',control=control)
# Print summary with p-values based on Satterthwaite denominator degrees of freedom
```

```
summary(m,ddf='Satterthwaite')

# Example for fitting a model without correlations in the random part
# (even for factor variables!)
# 1. Create explicit columns for factor variables
library(buildmer)
vowels <- cbind(vowels,model.matrix(~vowel,vowels))
# 2. Create formula with diagonal covariance structure
form <- diag(f1 ~ (vowel1+vowel2+vowel3+vowel4)*timepoint*following +
     ((vowel1+vowel2+vowel3+vowel4)*timepoint*following | participant) +
     (timepoint | word))
# 3. Convert formula to buildmer terms list
terms <- tabulate.formula(form)
# 4. Assign the different vowelN columns to identical blocks
terms[ 2: 5,'block'] <- 'same1'
terms[ 7:10,'block'] <- 'same2'
terms[12:15,'block'] <- 'same3'
terms[17:20,'block'] <- 'same4'
terms[22:25,'block'] <- 'same5'
terms[27:30,'block'] <- 'same6'
terms[32:35,'block'] <- 'same7'
terms[37:40,'block'] <- 'same8'
# 5. Directly pass the terms object to buildmer(), using the hidden 'dep' argument to specify
# the dependent variable
m <- buildmer(terms,data=vowels,dep='f1')
```

`buildmer-class` *The buildmer class*

Description

This is a simple convenience class that allows 'anova()' and 'summary()' calls to fall through to the underlying model object, while retaining buildmer's iteration history. If you need to use the final model for other things, such as prediction, access it through the 'model' slot of the buildmer class object.

Slots

`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()]

Examples

```
# Manually create a bare-bones buildmer object:
model <- lm(Sepal.Length ~ Petal.Length,iris)
p <- list(in.buildmer=FALSE)
library(buildmer)
bm <- mkBuildmer(model=model,p=p,anova=NULL,summary=NULL)
summary(bm)
```

`buildmultinom` *Use buildmer to perform stepwise elimination for* `multinom()` *models from package* `nnet`

Description

Use buildmer to perform stepwise elimination for `multinom()` models from package `nnet`

Usage

```
buildmultinom(formula, data = NULL, cl = NULL, direction = c("order",
  "backward"), crit = "LRT", include = NULL, 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 function `makeCluster()` from package `parallel` to use for parallelizing the evaluation of terms.
`direction`	Character string or vector indicating 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`	Character string or vector determining the criterion used to test terms for elimination. Possible options are `'LRT'` (likelihood-ratio test; this is the default), `'LL'` (use the raw -2 log likelihood), `'AIC'` (Akaike Information Criterion), and `'BIC'` (Bayesian Information Criterion).
`include`	A character vector of terms that will be kept in the model at all times. These do not need to be specified separately in the `formula` argument.
`calc.summary`	Logical indicating whether to also calculate the summary table for the final model after term elimination.
`quiet`	Logical indicating whether to suppress progress messages.
`...`	Additional options to be passed to `multinom()`.

See Also

buildmer()

Examples

```
library(buildmer)
options(contrasts = c("contr.treatment", "contr.poly"))
library(MASS)
example(birthwt)
bwt.mu <- buildmultinom(low ~ age*lwt*race*smoke,bwt)
```

---

conv *Test a model for convergence*

Description

Test a model for convergence

Usage

```
conv(model)
```

Arguments

model	The model object to test.

Value

Logical indicating whether the model converged.

Examples

```
library(buildmer)
library(lme4)
good1 <- lm(Reaction ~ Days,sleepstudy)
good2 <- lmer(Reaction ~ Days + (Days|Subject),sleepstudy)
bad <- lmer(Reaction ~ Days + (Days|Subject),sleepstudy,control=lmerControl(
            optimizer='bobyqa',optCtrl=list(maxfun=1)))
sapply(c(good1,good2,bad),conv)
```

`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

`x`	A model formula.

Value

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

Examples

```
# 1. Create explicit columns for factor variables
library(buildmer)
vowels <- cbind(vowels,model.matrix(~vowel,vowels))
# 2. Create formula with diagonal covariance structure
form <- diag(f1 ~ (vowel1+vowel2+vowel3+vowel4)*timepoint*following +
     ((vowel1+vowel2+vowel3+vowel4)*timepoint*following | participant) +
     (timepoint | word))
# 3. Convert formula to buildmer terms list
terms <- tabulate.formula(form)
# 4. Assign the different vowelN columns to identical blocks
terms[ 2: 5,'block'] <- 'same1'
terms[ 7:10,'block'] <- 'same2'
terms[12:15,'block'] <- 'same3'
terms[17:20,'block'] <- 'same4'
terms[22:25,'block'] <- 'same5'
terms[27:30,'block'] <- 'same6'
terms[32:35,'block'] <- 'same7'
terms[37:40,'block'] <- 'same8'
# 5. Directly pass the terms object to buildmer(), using the hidden 'dep' argument to specify
# the dependent variable

m <- buildmer(terms,data=vowels,dep='f1')
```

migrant *A very small data set from a pilot study on sound change.*

Description

A very small data set from a pilot study on sound change.

Usage

```
data(migrant)
```

Format

A standard data frame.

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.

Examples

```
library(buildmer)
remove.terms(Reaction ~ Days + (Days|Subject),'(Days|Subject)')
# illustration of the marginality checking mechanism:
remove.terms(Reaction ~ Days + (Days|Subject),'(1|Subject)') #refuses to remove the term
remove.terms(Reaction ~ Days + (Days|Subject),c('(Days|Subject)','(1|Subject)')) #also
          #refuses to remove the term, because marginality is checked before removal!
step1 <- remove.terms(Reaction ~ Days + (Days|Subject),'(Days|Subject)')
step2 <- remove.terms(step1,'(1|Subject)') #works
```

tabulate.formula *Parse a formula into a buildmer terms list*

Description

Parse a formula into a buildmer terms list

Usage

```
tabulate.formula(formula)
```

Arguments

formula	A formula.

Value

A buildmer terms list, which is just a normal data frame.

Examples

```
form <- diag(f1 ~ vowel*timepoint*following + ((vowel1+vowel2+vowel3+vowel4)*timepoint*
                  following|participant) + (timepoint|word))
tabulate.formula(form)
```

vowels *Vowel data from a pilot study.*

Description

Vowel data from a pilot study.

Usage

```
data(vowels)
```

Format

A standard data frame.

Index