

Package ‘reformulas’

October 28, 2025

Title Machinery for Processing Random Effect Formulas

Version 0.4.2

Description Takes formulas including random-effects components (formatted as in 'lme4', 'glmmTMB', etc.) and processes them. Includes various helper functions.

URL <https://github.com/bbolker/reformulas>

License GPL-3

Encoding UTF-8

Imports stats, methods, Matrix, Rdpack

RdMacros Rdpack

Suggests lme4, tinytest, glmmTMB

RoxygenNote 7.3.2.9000

NeedsCompilation no

Author Ben Bolker [aut, cre] (ORCID: <https://orcid.org/0000-0002-2127-0443>)

Maintainer Ben Bolker <bolker@mcmaster.ca>

Repository CRAN

Date/Publication 2025-10-28 06:10:17 UTC

Contents

anySpecial	2
expandDoubleVerts	2
expandGrpVar	3
findReTrmClasses	3
isNested	4
mkReTrms	4
nobars	6
no_specials	7
randint	8
RHSForm	8
subbars	9
Index	10

anySpecial	<i>Detect whether there are any 'specials' in a formula term</i>
------------	--

Description

Detect whether there are any 'specials' in a formula term

Usage

```
anySpecial(term, specials = findReTrmClasses(), fast = FALSE)
```

Arguments

term	formula term
specials	values to detect
fast	(logical) use quick (syntactic) test for presence of specials?

Value

logical value

Examples

```
## should only detect s as the head of a function, s(...)
anySpecial(~diag(1))
anySpecial(~diag)
anySpecial(~diag[[1]])
anySpecial(~diag[1])
anySpecial(~s)
anySpecial(~s(hello+goodbye,whatever))
```

expandDoubleVerts	<i>Expand terms with ' ' notation into separate ' ' terms</i>
-------------------	---

Description

From the right hand side of a formula for a mixed-effects model, expand terms with the double vertical bar operator into separate, independent random effect terms.

Usage

```
expandDoubleVerts(term)
```

Arguments

term	a mixed-model formula
------	-----------------------

Value

the modified term

See Also

[formula](#), [model.frame](#), [model.matrix](#).

Other utilities: [mkReTrms\(\)](#), [nobars\(\)](#), [subbars\(\)](#)

expandGrpVar	<i>apply operator expansion (e.g. $a/b \rightarrow a + a:b$) to a formula term</i>
--------------	---

Description

apply operator expansion (e.g. $a/b \rightarrow a + a:b$) to a formula term

Usage

```
expandGrpVar(f)
```

Arguments

f a language object (an atom of a formula)

Examples

```
expandGrpVar(quote(x*y))
expandGrpVar(quote(x/y))
```

findReTrmClasses	<i>list of specials – taken from enum.R</i>
------------------	---

Description

list of specials – taken from enum.R

Usage

```
findReTrmClasses()
```

isNested	<i>Is f1 nested within f2?</i>
----------	--------------------------------

Description

Does every level of f1 occur in conjunction with exactly one level of f2? The function is based on converting a triplet sparse matrix to a compressed column-oriented form in which the nesting can be quickly evaluated.

Usage

```
isNested(f1, f2)
```

Arguments

f1	factor 1
f2	factor 2

Value

TRUE if factor 1 is nested within factor 2

Examples

```
if (requireNamespace("lme4")) {
  data("Pastes", package = "lme4")
  with(Pastes, isNested(cask, batch)) ## => FALSE
  with(Pastes, isNested(sample, batch)) ## => TRUE
}
```

mkReTrms	<i>Create list of structures needed for models with random effects</i>
----------	--

Description

From the result of [findbars](#) applied to a model formula and the evaluation frame, create the model matrix, etc. associated with random-effects terms. See the description of the returned value for a detailed list.

Usage

```
mkReTrms(
  bars,
  fr,
  drop.unused.levels = TRUE,
  reorder.terms = TRUE,
  reorder.vars = FALSE,
  calc.lammdat = TRUE,
  sparse = NULL
)
```

Arguments

<code>bars</code>	a list of parsed random-effects terms
<code>fr</code>	a model frame in which to evaluate these terms
<code>drop.unused.levels</code>	(logical) drop unused factor levels?
<code>reorder.terms</code>	arrange random effects terms in decreasing order of number of groups (factor levels)?
<code>reorder.vars</code>	arrange columns of individual random effects terms in alphabetical order?
<code>calc.lammdat</code>	(logical) compute Lammdat and Lind components? (At present these components are needed for lme4 machinery but not for glmmTMB, and may be large in some cases; see Bates <i>et al.</i> 2015)
<code>sparse</code>	(logical) set up sparse model matrices?

Details

Lammdat, Lind, theta, lower are likely to be useful only for lme4; the other terms can be generally useful for constructing mixed-effect models

Value

	a list with components
<code>Zt</code>	transpose of the sparse model matrix for the random effects
<code>Ztlist</code>	list of components of the transpose of the random-effects model matrix, separated by random-effects term
<code>Lammdat</code>	transpose of the sparse relative covariance factor
<code>Lind</code>	an integer vector of indices determining the mapping of the elements of the theta to the "x" slot of Lammdat
<code>theta</code>	initial values of the covariance parameters
<code>lower</code>	lower bounds on the covariance parameters
<code>flist</code>	list of grouping factors used in the random-effects terms
<code>cnms</code>	a list of column names of the random effects according to the grouping factors

Gp	a vector indexing the association of elements of the conditional mode vector with random-effect terms; if nb is the vector of numbers of conditional modes per term (i.e. number of groups times number of effects per group), Gp is <code>c(0, cumsum(nb))</code> (and conversely nb is <code>diff(Gp)</code>)
nl	names of the terms (in the same order as Zt, i.e. reflecting the <code>reorder.terms</code> argument)
ord	an integer vector giving the relationship between the order of the terms in the formula and the terms in the final object (which are ordered by the number of levels in the grouping variable, if <code>reorder.terms</code> is TRUE)

References

Bates D, Mächler M, Bolker B, Walker S (2015). “Fitting Linear Mixed-Effects Models Using lme4.” *Journal of Statistical Software*, **67**(1), 1–48. doi:[10.18637/jss.v067.i01](https://doi.org/10.18637/jss.v067.i01).

See Also

Other utilities: [expandDoubleVerts\(\)](#), [nobars\(\)](#), [subbars\(\)](#)

Examples

```
## (silly/impractical formula, for illustration only)
form <- mpg ~ 1 + (1|gear) + (factor(cyl)|gear) + (1 + hp | carb)
fr <- model.frame(subbars(form), data = mtcars)
rterms <- mkReTrms(findbars(form), fr)
names(rterms)
## block sizes (latent variables per block) of each term
(nperblock <- lengths(rterms$cnms))
## latent variables per term
(nperterm <- diff(rterms$Gp))
with(rterms, identical(unname(nl*nperblock), nperterm))
## illustrate reordering of terms
dd <- expand.grid(a = 1:7, b = 1:3, c = 1:5, d = 1:9)
dd$y <- 1
form2 <- y ~ 1 + (1|a) + (1|b) + (1|c) + (1|d)
rterms2 <- mkReTrms(findbars(form2), dd, reorder.terms = TRUE)
## reorder elements into original formula order
with(rterms2, cnms[order(ord)])
## reorder splitForm output to match mkReTrms components
ss <- splitForm(form2)
ss$reTrmFormulas[rterms2$ord]
```

nobars

Omit terms separated by vertical bars in a formula

Description

Remove the random-effects terms from a mixed-effects formula, thereby producing the fixed-effects formula.

Usage

```
nobars(term)
```

```
nobars_(term)
```

Arguments

term the right-hand side of a mixed-model formula

Value

the fixed-effects part of the formula

Note

This function is called recursively on individual terms in the model, which is why the argument is called `term` and not a name like `form`, indicating a formula.

See Also

[formula](#), [model.frame](#), [model.matrix](#).

Other utilities: [expandDoubleVerts\(\)](#), [mkReTrms\(\)](#), [subbars\(\)](#)

Examples

```
nobars(Reaction ~ Days + (Days|Subject)) ## => Reaction ~ Days
```

no_specials	<i>Drop 'specials' from a formula</i>
-------------	---------------------------------------

Description

Drop 'specials' from a formula

Usage

```
no_specials(term, specials = c("|", "||", "s"))
```

Arguments

term a term or formula or list thereof

specials function types to drop

Value

a call or language object (or list) with specials removed

Examples

```
no_specials(findbars_x(~ 1 + s(x) + (f|g) + diag(x|y)))
no_specials(~us(f|g))
```

randint	<i>Remove all random slopes from a formula, while retaining random intercepts.</i>
---------	--

Description

Remove all random slopes from a formula, while retaining random intercepts.

Usage

```
randint(form)
```

Arguments

form A formula

Value

The new formula

Examples

```
f <- ~ 1 + a + b + (a | f) + (1 + a | g) + (a + b | h) + (1 + a + b | i)
randint(f)
```

RHSForm	<i>extract right-hand side of a formula</i>
---------	---

Description

extract right-hand side of a formula

Usage

```
RHSForm(form, as.form = FALSE)
```

Arguments

form a formula object
as.form (logical) return a formula (TRUE) or as a call/symbolic object (FALSE) ?

Value

a language object

Examples

```
RHSForm(y ~ x + (1|g))
```

subbars	<i>"Substitute bars"</i>
---------	--------------------------

Description

Substitute the '+' function for the '|' and '||' function in a mixed-model formula. This provides a formula suitable for the current model.frame function.

Usage

```
subbars(term)
```

Arguments

term	a mixed-model formula
------	-----------------------

Value

the formula with all | and || operators replaced by +

Note

This function is called recursively on individual terms in the model, which is why the argument is called term and not a name like form, indicating a formula.

See Also

[formula](#), [model.frame](#), [model.matrix](#).

Other utilities: [expandDoubleVerts\(\)](#), [mkReTrms\(\)](#), [nobars\(\)](#)

Examples

```
subbars(Reaction ~ Days + (Days|Subject)) ## => Reaction ~ Days + (Days + Subject)
```

Index

* **models**

- expandDoubleVerts, [2](#)
- nobars, [6](#)
- subbars, [9](#)

* **utilities**

- expandDoubleVerts, [2](#)
- mkReTrms, [4](#)
- nobars, [6](#)
- subbars, [9](#)

anySpecial, [2](#)

expandDoubleVerts, [2](#), [6](#), [7](#), [9](#)

expandGrpVar, [3](#)

findbars, [4](#)

findReTrmClasses, [3](#)

formula, [3](#), [7](#), [9](#)

isNested, [4](#)

mkReTrms, [3](#), [4](#), [7](#), [9](#)

model.frame, [3](#), [7](#), [9](#)

model.matrix, [3](#), [7](#), [9](#)

no_specials, [7](#)

nobars, [3](#), [6](#), [6](#), [9](#)

nobars_(nobars), [6](#)

randint, [8](#)

RHSForm, [8](#)

subbars, [3](#), [6](#), [7](#), [9](#)