Package 'glmmTMB'

July 3, 2018

Title Generalized Linear Mixed Models using Template Model Builder

Version 0.2.2.0

Description Fit linear and generalized linear mixed models with various extensions, including zero-inflation. The models are fitted using maximum likelihood estimation via 'TMB' (Template Model Builder). Random effects are assumed to be Gaussian on the scale of the linear predictor and are integrated out using the Laplace approximation. Gradients are calculated using automatic differentiation.

License AGPL-3

Depends R (>= 3.2.0)

Imports methods, TMB (>= 1.7.13), lme4 (>= 1.1-10), Matrix, nlme

LinkingTo TMB, RcppEigen

Suggests knitr, rmarkdown, testthat, MASS, lattice, ggplot2, mlmRev, bbmle (>= 1.0.19), pscl, MCMCpack, coda, reshape2, plyr

VignetteBuilder knitr

URL https://github.com/glmmTMB

LazyData TRUE

BugReports https://github.com/glmmTMB/glmmTMB/issues

RoxygenNote 6.0.1.9000

NeedsCompilation yes

Author Arni Magnusson [aut],
Hans Skaug [aut],
Anders Nielsen [aut],
Casper Berg [aut],
Kasper Kristensen [aut],
Martin Maechler [aut],
Koen van Bentham [aut],
Ben Bolker [aut],
Mollie Brooks [aut, cre]

Maintainer Mollie Brooks <mollieebrooks@gmail.com>

Encoding UTF-8
Repository CRAN
Date/Publication 2018-07-03 04:10:16 UTC

R topics documented:

`confint.glmmTMB` *Calculate confidence intervals*

Description

Calculate confidence intervals

Usage

```
## S3 method for class 'glmmTMB'
confint(object, parm, level = 0.95, method = c("wald",
  "Wald", "profile", "uniroot"), component = c("all", "cond", "zi", "other"),
  estimate = TRUE, parallel = c("no", "multicore", "snow"),
  ncpus = getOption("profile.ncpus", 1L), cl = NULL, ...)
```

Arguments

object	glmmTMB fitted object.
parm	Specification of a parameter subset *after* component subset has been applied.
level	Confidence level.
method	'wald', 'profile', or 'uniroot': see Details function)
component	Which of the three components 'cond', 'zi' or 'other' to select. Default is to select 'all'.
estimate	(logical) add a third column with estimate ?
parallel	method (if any) for parallel computation
ncpus	number of CPUs/cores to use for parallel computation
cl	cluster to use for parallel computation
...	arguments may be passed to profile.merMod or tmbroot

Details

Available methods are

wald These intervals are based on the standard errors calculated for parameters on the scale of their internal parameterization depending on the family. Derived quantities such as standard deviation parameters and dispersion parameters are backtransformed. It follows that confidence intervals for these derived quantities are asymmetric.

profile This method computes a likelihood profile for the specified parameter(s) using profile.glmmTMB; fits a spline function to each half of the profile; and inverts the function to find the specified confidence interval.

uniroot This method uses the uniroot function to find critical values of one-dimensional profile functions for each specified parameter.

Examples

```
data(sleepstudy, package="lme4")
model <- glmmTMB(Reaction ~ Days + (1|Subject), sleepstudy)
confint(model)
## Not run:
confint(model,parm=1,method="profile")

## End(Not run)
```

epil2 *Seizure Counts for Epileptics - Extended*

Description

Extended version of the `epil` dataset of the **MASS** package. The three transformed variables `Visit`, `Base`, and `Age` used by Booth et al. (2003) have been added to `epil`.

Usage

```
epil2
```

Format

A data frame with 236 observations on the following 12 variables:

`y` an integer vector.

`trt` a factor with levels `"placebo"` and `"progabide"`.

`base` an integer vector.

`age` an integer vector.

`V4` an integer vector.

`subject` an integer vector.

`period` an integer vector.

`lbase` a numeric vector.

`lage` a numeric vector.

Visit `(rep(1:4,59) - 2.5) / 5`.

Base `log(base/4)`.

Age `log(age)`.

References

Booth, J.G., G. Casella, H. Friedl, and J.P. Hobert. (2003) Negative binomial loglinear mixed models. *Statistical Modelling* **3**, 179–191.

Examples

```
epil2$subject <- factor(epil2$subject)
op <- options(digits=3)
(fm <- glmmTMB(y ~ Base*trt + Age + Visit + (Visit|subject),
              data=epil2, family=nbinom2))
meths <- methods(class = class(fm))
if((Rv <- getRversion()) > "3.1.3") {
    (funs <- attr(meths, "info")[, "generic"])
    for(F in funs[is.na(match(funs, "getME"))]) {
```

```
        cat(sprintf("%s:\n-----\n", F))
        r <- tryCatch( get(F)(fm), error=identity)
        if (inherits(r, "error")) cat("** Error:", r$message,"\n")
        else tryCatch( print(r) )
        cat(sprintf("---end{%s}--------------\n\n", F))
    }
}
options(op)
```

`findReTrmClasses` *list of specials – taken from enum.R*

Description

list of specials – taken from enum.R

Usage

```
findReTrmClasses()
```

`fixef` *Extract fixed-effects estimates*

Description

Extract the fixed-effects estimates

Usage

```
## S3 method for class 'glmmTMB'
fixef(object, ...)
```

Arguments

`object`	any fitted model object from which fixed effects estimates can be extracted.
`...`	optional additional arguments. Currently none are used in any methods.

Details

Extract the estimates of the fixed-effects parameters from a fitted model.

Value

a named, numeric vector of fixed-effects estimates.

Examples

```
data(sleepstudy, package = "lme4")
fixef(glmmTMB(Reaction ~ Days + (1|Subject) + (0+Days|Subject), sleepstudy))
```

`formatVC` *Format the 'VarCorr' Matrix of Random Effects*

Description

"format()" the 'VarCorr' matrix of the random effects – for print()ing and show()ing

Usage

```
formatVC(varcor, digits = max(3, getOption("digits") - 2),
  comp = "Std.Dev.", formatter = format, useScale = attr(varcor, "useSc"),
  ...)
```

Arguments

`varcor`	a `VarCorr` (-like) matrix with attributes.
`digits`	the number of significant digits.
`comp`	character vector of length one or two indicating which columns out of "Variance" and "Std.Dev." should be shown in the formatted output.
`formatter`	the `function` to be used for formatting the standard deviations and or variances (but *not* the correlations which (currently) are always formatted as "0.nnn"
`useScale`	whether to report a scale parameter (e.g. residual standard deviation)
`...`	optional arguments for `formatter(*)` in addition to the first (numeric vector) and `digits`.

Value

a character matrix of formatted VarCorr entries from `varc`.

`getCapabilities` *List model options that glmmTMB knows about*

Description

List model options that glmmTMB knows about

Usage

```
getCapabilities(what = "all", check = FALSE)
```

Arguments

what	(character) which type of model structure to report on ("all","family","link","covstruct")
check	(logical) do brute-force checking to test whether families are really implemented (only available for what="family")

Value

if check==FALSE, returns a vector of the names (or a list of name vectors) of allowable entries; if check==TRUE, returns a logical vector of working families

Note

these are all the options that are *defined* internally; they have not necessarily all been *implemented* (FIXME!)

---

getME.glmmTMB *Extract or Get Generalize Components from a Fitted Mixed Effects Model*

Description

Extract or Get Generalize Components from a Fitted Mixed Effects Model

Usage

```
## S3 method for class 'glmmTMB'
getME(object, name = c("X", "Xzi", "Z", "Zzi", "Xd",
  "theta"), ...)
```

Arguments

object	a fitted glmmTMB object
name	of the component to be retrieved
...	ignored, for method compatibility

See Also

getME Get generic and re-export:

getReStruc *Calculate random effect structure Calculates number of random effects, number of parameters, blocksize and number of blocks. Mostly for internal use.*

Description

Calculate random effect structure Calculates number of random effects, number of parameters, blocksize and number of blocks. Mostly for internal use.

Usage

```
getReStruc(reTrms, ss = NULL)
```

Arguments

reTrms	random-effects terms list
ss	a character string indicating a valid covariance structure. Must be one of names(glmmTMB:::.valid_covs default is to use an unstructured variance-covariance matrix ("us") for all blocks).

Value

a list

blockNumTheta	number of variance covariance parameters per term
blockSize	size (dimension) of one block
blockReps	number of times the blocks are repeated (levels)
covCode	structure code

Examples

```
data(sleepstudy, package="lme4")
rt <- lme4::lFormula(Reaction~Days+(1|Subject)+(0+Days|Subject),
                     sleepstudy)$reTrms
rt2 <- lme4::lFormula(Reaction~Days+(Days|Subject),
                     sleepstudy)$reTrms
getReStruc(rt)
```

getXReTrms *Create X and random effect terms from formula*

Description

Create X and random effect terms from formula

Usage

```
getXReTrms(formula, mf, fr, ranOK = TRUE, type = "")
```

Arguments

formula	current formula, containing both fixed & random effects
mf	matched call
fr	full model frame
ranOK	random effects allowed here?
type	label for model type

Value

a list composed of

X	design matrix for fixed effects
Z	design matrix for random effects
reTrms	output from mkReTrms from **lme4**
offset	offset vector, or vector of zeros if offset not specified

glmmTMB *Fit models with TMB*

Description

Fit models with TMB

Usage

```
glmmTMB(formula, data = NULL, family = gaussian(), ziformula = ~0,
  dispformula = ~1, weights = NULL, offset = NULL, na.action = na.fail,
  se = TRUE, verbose = FALSE, doFit = TRUE, control = glmmTMBControl(),
  REML = FALSE)
```

Arguments

formula	combined fixed and random effects formula, following lme4 syntax
data	data frame
family	a family function, a character string naming a family function, or the result of a call to a family function family (variance/link function) information; see family for generic discussion of families or family_glmmTMB for details of glmmTMB-specific families.
ziformula	a *one-sided* (i.e., no response variable) formula for zero-inflation combining fixed and random effects: the default ~0 specifies no zero-inflation. Specifying ~. will set the right-hand side of the zero-inflation formula identical to the right-hand side of the main (conditional effects) formula; terms can also be added or subtracted. **Offset terms will automatically be dropped from the conditional effects formula when using** ~. The zero-inflation model uses a logit link.
dispformula	a *one-sided* formula for dispersion containing only fixed effects: the default ~1 specifies the standard dispersion given any family. The argument is ignored for families that do not have a dispersion parameter. For an explanation of the dispersion parameter for each family, see (sigma). The dispersion model uses a log link. In Gaussian mixed models, dispformula=~0 fixes the parameter to be 0, forcing variance into the random effects.
weights	weights, as in glm. Not automatically scaled to have sum 1.
offset	offset for conditional model (only):
na.action	how to handle missing values (see na.action and model.frame); from lm, "The default is set by the na.action setting of options, and is na.fail if that is unset. The 'factory-fresh' default is na.omit."
se	whether to return standard errors
verbose	logical indicating if some progress indication should be printed to the console.
doFit	whether to fit the full model, or (if FALSE) return the preprocessed data and parameter objects, without fitting the model
control	control parameters; see glmmTMBControl.
REML	Logical; Use REML estimation rather than maximum likelihood.

Details

- binomial models with more than one trial (i.e., not binary/Bernoulli) can either be specified in the form prob ~ ..., weights = N or in the more typical two-column matrix (cbind(successes,failures)~...) form.
- Behavior of REML=TRUE for Gaussian responses matches lme4::lmer). It may also be useful in some cases with non-Gaussian responses (Millar 2011). Simulations should be done first to verify.
- Because the df.residual method for glmmTMB currently counts the dispersion parameter, one would need to multiply by sqrt(nobs(fit)/(1+df.residual(fit))) when comparing with lm ...
- by default, vector-valued random effects are fitted with unstructured (general positive definite) variance-covariance matrices. Structured variance-covariance matrices can be specified in the form struc(terms|group), where struc is one of

  - diag (diagonal, heterogeneous variance)
  - ar1 (autoregressive order-1, homogeneous variance)
  - cs (compound symmetric, heterogeneous variance)
  - ou (* Ornstein-Uhlenbeck, homogeneous variance)
  - exp (* exponential autocorrelation)
  - gau (* Gaussian autocorrelation)
  - mat (* Matérn process correlation)
  - toep (* Toeplitz)

  (note structures marked with * are experimental/untested)

- For backward compatibility, the family argument can also be specified as a list comprising the name of the distribution and the link function (e.g. 'list(family="binomial", link="logit")'). However, **this alternatives is now deprecated** (it produces a warning and will be removed at some point in the future). Furthermore, certain capabilities such as Pearson residuals or predictions on the data scale will only be possible if components such as variance and linkfun are present (see family).

References

- Millar, Russell B. Maximum Likelihood Estimation and Inference: With Examples in R, SAS and ADMB. John Wiley & Sons, 2011.

Examples

```
(m1 <- glmmTMB(count~ mined + (1|site),
  zi=~mined,
  family=poisson, data=Salamanders))
summary(m1)

## Zero-inflated negative binomial model
(m2 <- glmmTMB(count~spp + mined + (1|site),
  zi=~spp + mined,
  family=nbinom2, Salamanders))

## Hurdle Poisson model
(m3 <- glmmTMB(count~spp + mined + (1|site),
  zi=~spp + mined,
  family=truncated_poisson, Salamanders))

## Binomial model
data(cbpp, package="lme4")
(tmbm1 <- glmmTMB(cbind(incidence, size-incidence) ~ period + (1 | herd),
               data=cbpp, family=binomial))

## Dispersion model
sim1=function(nfac=40, nt=100, facsd=.1, tsd=.15, mu=0, residsd=1)
{
  dat=expand.grid(fac=factor(letters[1:nfac]), t= 1:nt)
  n=nrow(dat)
  dat$REfac=rnorm(nfac, sd= facsd)[dat$fac]
  dat$REt=rnorm(nt, sd= tsd)[dat$t]
```

```
  dat$x=rnorm(n, mean=mu, sd=residsd) + dat$REfac + dat$REt
  return(dat)
}
set.seed(101)
d1 = sim1(mu=100, residsd =10)
d2 = sim1(mu=200, residsd =5)
d1$sd="ten"
d2$sd="five"
dat = rbind(d1, d2)
m0 = glmmTMB(x~sd+(1|t), dispformula=~sd, dat)
fixef(m0)$disp
c(log(5^2), log(10^2)-log(5^2)) #expected dispersion model coefficients
```

`glmmTMBControl` *Control parameters for glmmTMB optimization*

Description

Control parameters for glmmTMB optimization

Usage

```
glmmTMBControl(optCtrl = list(iter.max = 300, eval.max = 400),
  profile = FALSE, collect = FALSE)
```

Arguments

`optCtrl`	Passed as argument `control` to `nlminb`.
`profile`	Logical; Experimental option to improve speed and robustness when a model has many fixed effects
`collect`	Logical; Experimental option to improve speed by recognizing duplicated observations.

Details

The general non-linear optimizer `nlminb` is used by `glmmTMB` for parameter estimation. It may sometimes be necessary to tweak some tolerances in order to make a model converge. For instance, the warning 'iteration limit reached without convergence' may be fixed by increasing the number of iterations using something like

`glmmTMBControl(optCtrl=list(iter.max=1e3,eval.max=1e3))`.

The argument `profile` allows `glmmTMB` to use some special properties of the optimization problem in order to speed up estimation in cases with many fixed effects. Enable this option using

`glmmTMBControl(profile=TRUE)`.

Control parameters may depend on the model specification, because each control component is evaluated inside `TMBStruc`, the output of `mkTMBStruc`. To specify that `profile` should be enabled for more than 5 fixed effects one can use

`glmmTMBControl(profile=quote(length(parameters$beta)>=5))`.

nbinom2 *Family functions for glmmTMB*

Description

Family functions for glmmTMB

Usage

```
nbinom2(link = "log")

nbinom1(link = "log")

compois(link = "log")

truncated_compois(link = "log")

genpois(link = "log")

truncated_genpois(link = "log")

truncated_poisson(link = "log")

truncated_nbinom2(link = "log")

truncated_nbinom1(link = "log")

beta_family(link = "logit")

betabinomial(link = "logit")

tweedie(link = "log")
```

Arguments

link	(character) link function for the conditional mean ("log", "logit", "probit", "inverse", "cloglog", or "identity")

Details

If specified, the dispersion model uses a log link. Denoting the dispersion parameter as phi=exp(eta) (where eta is the linear predictor from the dispersion model) and the predicted mean as mu:

gaussian (from base R): constant variance=phi

Gamma (from base R) phi is the shape parameter, i.e variance=mu*phi

nbinom2 variance increases quadratically with the mean (Hardin & Hilbe 2007), i.e. variance=mu*(1+mu/phi)

nbinom1 variance increases linearly with the mean (Hardin & Hilbe 2007), i.e. variance=mu*(1+phi)

compois is the Conway-Maxwell Poisson parameterized with the exact mean which differs from the COMPoissonReg package (Sellers & Lotze 2015)

genpois is the generalized Poisson distribution

beta follows the parameterization of Ferrari and Cribari-Neto (2004) and the `betareg` package, i.e. variance=mu*(1-mu)/(1+phi)

Value

returns a list with (at least) components

`family`	length-1 character vector giving the family name
`link`	length-1 character vector specifying the link function
`variance`	a function of either 1 (mean) or 2 (mean and dispersion parameter) arguments giving the predicted variance

References

- Ferrari SLP, Cribari-Neto F (2004). "Beta Regression for Modelling Rates and Proportions." *J. Appl. Stat.* 31(7), 799-815.
- Hardin JW & Hilbe JM (2007). "Generalized linear models and extensions." Stata Press.
- Sellers K & Lotze T (2015). "COMPoissonReg: Conway-Maxwell Poisson (COM-Poisson) Regression". R package version 0.3.5. https://CRAN.R-project.org/package=COMPoissonReg

---

`numFactor` *Factor with numeric interpretable levels.*

Description

Create a factor with numeric interpretable factor levels.

Usage

```
numFactor(x, ...)

parseNumLevels(levels)
```

Arguments

`x`	Vector, matrix or data.frame that constitute the coordinates.
`...`	Additional vectors, matrices or data.frames that constitute the coordinates.
`levels`	Character vector to parse into numeric values.

Details

Some glmmTMB covariance structures require extra information, such as temporal or spatial coordinates. numFactor allows to associate such extra information as part of a factor via the factor levels. The original numeric coordinates are recoverable without loss of precision using the function parseNumLevels. Factor levels are sorted coordinate wise from left to right: first coordinate is fastest running.

Value

Factor with specialized coding of levels.

Examples

```
## 1D example
numFactor(sample(1:5,20,TRUE))
## 2D example
coords <- cbind( sample(1:5,20,TRUE), sample(1:5,20,TRUE) )
(f <- numFactor(coords))
parseNumLevels(levels(f)) ## Sorted
## Used as part of a model.matrix
model.matrix( ~f )
## parseNumLevels( colnames(model.matrix( ~f )) )
## Error: 'Failed to parse numeric levels: (Intercept)'
parseNumLevels( colnames(model.matrix( ~ f-1 )) )
```

Owls *Begging by Owl Nestlings*

Description

Begging by owl nestlings

Usage

```
data(Owls)
```

Format

The Owls data set is a data frame with 599 observations on the following variables:

Nest a factor describing individual nest locations

FoodTreatment (factor) food treatment: Deprived or Satiated

SexParent (factor) sex of provisioning parent: Female or Male

ArrivalTime a numeric vector

SiblingNegotiation a numeric vector

BroodSize brood size

NegPerChick number of negotations per chick

Note

Access to data kindly provided by Alain Zuur

Source

Roulin, A. and L. Bersier (2007) Nestling barn owls beg more intensely in the presence of their mother than in the presence of their father. *Animal Behaviour* **74** 1099–1106. http://www.sciencedirect.com/science/article/B6W9W-4PK8B6H-8/2/e43cfbaad4dc0bb2207adfc54a460c89; http://www.highstat.com/Books/Book2/ZuurDataMixedModelling.zip

References

Zuur, A. F., E. N. Ieno, N. J. Walker, A. A. Saveliev, and G. M. Smith (2009) *Mixed Effects Models and Extensions in Ecology with R*; Springer.

Examples

```
data(Owls, package = "glmmTMB")
require("lattice")
bwplot(reorder(Nest,NegPerChick) ~ NegPerChick | FoodTreatment:SexParent,
       data=Owls)
dotplot(reorder(Nest,NegPerChick) ~ NegPerChick| FoodTreatment:SexParent,
        data=Owls)
## Not run:
## Fit negative binomial model with "constant" Zero Inflation :
owls_nb1 <- glmmTMB(SiblingNegotiation ~ FoodTreatment*SexParent +
                                    (1|Nest)+offset(log(BroodSize)),
              family = nbinom1(), zi = ~1, data=Owls)
owls_nb1_bs <- update(owls_nb1,
                      . ~ . - offset(log(BroodSize)) + log(BroodSize))
fixef(owls_nb1_bs)

## End(Not run)
```

---

predict.glmmTMB *prediction*

Description

prediction

Usage

```
## S3 method for class 'glmmTMB'
predict(object, newdata = NULL, se.fit = FALSE, re.form,
  allow.new.levels = FALSE, type = c("link", "response", "conditional",
  "zprob", "zlink"), zitype = NULL, na.action = na.pass, debug = FALSE,
  ...)
```

Arguments

object	a glmmTMB object
newdata	new data for prediction
se.fit	return the standard errors of the predicted values?
re.form	(not yet implemented) specify which random effects to condition on when predicting
allow.new.levels	allow previously unobserved levels in random-effects variables? see details.
type	Denoting mu as the mean of the conditional distribution and p as the zero-inflation probability, the possible choices are: **"link"** conditional mean on the scale of the link function, or equivalently the linear predictor of the conditional model **"response"** expected value; this is $mu * (1 - p)$ for zero-inflated models and mu otherwise **"conditional"** mean of the conditional response; mu for all models (i.e., synonymous with "response" in the absence of zero-inflation **"zprob"** the probability of a structural zero (gives an error for non-zero-inflated models) ##' **"zilink"** predicted zero-inflation probability on the scale of the logit link function
zitype	deprecated: formerly used to specify type of zero-inflation probability. Now synonymous with type
na.action	how to handle missing values in newdata (see na.action); the default (na.pass) is to predict NA
debug	(logical) return the TMBStruc object that will be used internally for debugging?
...	unused - for method compatibility

Details

Prediction of new random effect levels is possible as long as the model specification (fixed effects and parameters) is kept constant. However, to ensure intentional usage, a warning is triggered if allow.new.levels=FALSE (the default).

Examples

```
data(sleepstudy,package="lme4")
g0 <- glmmTMB(Reaction~Days+(Days|Subject),sleepstudy)
predict(g0, sleepstudy)
## Predict new Subject
nd <- sleepstudy[1,]
nd$Subject <- "new"
predict(g0, newdata=nd, allow.new.levels=TRUE)
```

print.VarCorr.glmmTMB *Printing The Variance and Correlation Parameters of a* glmmTMB

Description

Printing The Variance and Correlation Parameters of a glmmTMB

Usage

```
## S3 method for class 'VarCorr.glmmTMB'
print(x, digits = max(3, getOption("digits") - 2),
  comp = "Std.Dev.", formatter = format, ...)
```

Arguments

x	a result of VarCorr(<glmmTMB>).
digits	number of significant digits to use.
comp	a string specifying the component to format and print.
formatter	a function.
...	optional further arguments, passed the next print method.

profile.glmmTMB *Compute likelihood profiles for a fitted model*

Description

Compute likelihood profiles for a fitted model

Usage

```
## S3 method for class 'glmmTMB'
profile(fitted, parm = NULL, level_max = 0.99, npts = 8,
  stepfac = 1/4, stderr = NULL, trace = FALSE, parallel = c("no",
  "multicore", "snow"), ncpus = getOption("profile.ncpus", 1L), cl = NULL,
  ...)

## S3 method for class 'profile.glmmTMB'
confint(object, parm = NULL, level = 0.95, ...)
```

Arguments

fitted	a fitted glmmTMB object
parm	which parameters to profile, specified • by index (position) • by name (matching the row/column names of vcov(object,full=TRUE)) • as "theta_" (random-effects variance-covariance parameters) or "beta_" (conditional and zero-inflation parameters)
level_max	maximum confidence interval target for profile
npts	target number of points in (each half of) the profile (*approximate*)
stepfac	initial step factor (fraction of estimated standard deviation)
stderr	standard errors to use as a scaling factor when picking step sizes to compute the profile; by default (if stderr is NULL, or NA for a particular element), uses the estimated (Wald) standard errors of the parameters
trace	print tracing information? If trace=FALSE or 0, no tracing; if trace=1, print names of parameters currently being profiled; if trace>1, turn on tracing for the underlying tmbprofile function
parallel	method (if any) for parallel computation
ncpus	number of CPUs/cores to use for parallel computation
cl	cluster to use for parallel computation
...	additional arguments passed to tmbprofile
object	a fitted profile (profile.glmmTMB) object
level	confidence level

Details

Fits natural splines separately to the points from each half of the profile for each specified parameter (i.e., values above and below the MLE), then finds the inverse functions to estimate the endpoints of the confidence interval

Value

An object of class profile.glmmTMB, which is also a data frame, with columns .par (parameter being profiled), .focal (value of focal parameter), value (negative log-likelihood).

Examples

```
## Not run:
m1 <- glmmTMB(count~ mined + (1|site),
       zi=~mined, family=poisson, data=Salamanders)
salamander_prof1 <- profile(m1, parallel="multicore",
                            ncpus=2, trace=1)
## testing
salamander_prof1 <- profile(m1, trace=1,parm=1)
salamander_prof1M <- profile(m1, trace=1,parm=1, npts = 4)
salamander_prof2 <- profile(m1, parm="theta_")
```

```
## End(Not run)
salamander_prof1 <- readRDS(system.file("example_files","salamander_prof1.rds",package="glmmTMB"))
if (require("ggplot2")) {
    ggplot(salamander_prof1,aes(.focal,sqrt(value))) +
        geom_point() + geom_line()+
        facet_wrap(~.par,scale="free_x")+
    geom_hline(yintercept=1.96,linetype=2)
}
salamander_prof1 <- readRDS(system.file("example_files","salamander_prof1.rds",package="glmmTMB"))
confint(salamander_prof1)
confint(salamander_prof1,level=0.99)
```

`ranef.glmmTMB` *Extract Random Effects*

Description

Generic function to extract random effects from `glmmTMB` models, both for the conditional model and zero inflation.

Usage

```
## S3 method for class 'glmmTMB'
ranef(object, ...)
```

Arguments

`object`	a `glmmTMB` model.
`...`	some methods for this generic function require additional arguments.

Value

Object of class `ranef.glmmTMB` with two components:

`cond`	a list of data frames, containing random effects for the conditional model.
`zi`	a list of data frames, containing random effects for the zero inflation.

Note

When a model has no zero inflation, the default behavior of `ranef` is to simplify the printed format of the random effects. To show the full list structure, run `print(ranef(model), simplify=FALSE)`. In all cases, the full list structure is used to access the data frames (see example).

See Also

`fixef.glmmTMB`.

Examples

```
data(sleepstudy, package="lme4")
model <- glmmTMB(Reaction ~ Days + (1|Subject), sleepstudy)
ranef(model)
print(ranef(model), simplify=FALSE)
ranef(model)$cond$Subject
```

`residuals.glmmTMB` *Compute residuals for a glmmTMB object*

Description

Compute residuals for a glmmTMB object

Usage

```
## S3 method for class 'glmmTMB'
residuals(object, type = c("response", "pearson"), ...)
```

Arguments

`object`	a "glmmTMB" object
`type`	(character) residual type
`...`	ignored, for method compatibility

`Salamanders` *Repeated counts of salamanders in streams*

Description

A dataset containing counts of salamanders with site covariates and sampling covariates. Each of 23 sites were sampled 4 times. When using this data, please cite Price et al. (2016) as well as the Dryad data package (Price et al. 2015).

Usage

```
data(Salamanders)
```

Format

A data frame with 644 observations on the following 10 variables:

site name of a location where repeated samples were taken

mined factor indicating whether the site was affected by mountian top removal coal mining

cover amount of cover objects in the stream (scaled)

sample repeated sample

DOP Days since precipitation (scaled)

Wtemp water temperature (scaled)

DOY day of year (scaled)

spp abreviated species name, possibly also life stage

count number of observed salamanders

References

Price SJ, Muncy BL, Bonner SJ, Drayer AN, Barton CD (2016) Effects of mountaintop removal mining and valley filling on the occupancy and abundance of stream salamanders. *Journal of Applied Ecology* **53** 459–468. http://dx.doi.org/10.1111/1365-2664.12585

Price SJ, Muncy BL, Bonner SJ, Drayer AN, Barton CD (2015) Data from: Effects of mountaintop removal mining and valley filling on the occupancy and abundance of stream salamanders. *Dryad Digital Repository*. http://dx.doi.org/10.5061/dryad.5m8f6

Examples

```
require("glmmTMB")
data(Salamanders)

zipm3 = glmmTMB(count~spp * mined + (1|site), zi=~spp * mined, Salamanders, family="poisson")
```

---

sigma.glmmTMB *Extract residual standard deviation or dispersion parameter*

Description

For Gaussian models, `sigma` returns the value of the residual standard deviation; for other families, it returns the dispersion parameter, *however it is defined for that particular family*. See details for each family below.

Usage

```
## S3 method for class 'glmmTMB'
sigma(object, ...)
```

Arguments

`object`	a "glmmTMB" fitted object
`...`	(ignored; for method compatibility)

Details

The value returned varies by family:

gaussian returns the *maximum likelihood* estimate of the standard deviation (i.e., smaller than the results of `sigma(lm(...))` by a factor of (n-1)/n)

nbinom1 returns an overdispersion parameter (usually denoted α as in Hardin and Hilbe (2007)): such that the variance equals $\mu(1+\alpha)$.

nbinom2 returns an overdispersion parameter (usually denoted θ or k); in contrast to most other families, larger θ corresponds to a *lower* variance which is $\mu(1+\mu/\theta)$.

Gamma Internally, glmmTMB fits Gamma responses by fitting a mean and a shape parameter; sigma is estimated as (1/sqrt(shape)), which will typically be close (but not identical to) that estimated by `stats:::sigma.default`, which uses sqrt(deviance/df.residual)

beta returns the value of ϕ, where the conditional variance is $\mu(1-\mu)/(1+\phi)$ (i.e., increasing ϕ decreases the variance.) This parameterization follows Ferrari and Cribari-Neto (2004) (and the `betareg` package):

betabinomial This family uses the same parameterization (governing the Beta distribution that underlies the binomial probabilities) as `beta`.

genpois returns the value of ϕ, where the variance is $\mu\phi$

compois returns the value of $1/\nu$, When $\nu = 1$, compois is equivalent to the Poisson distribution. There is no closed form equation for the variance, but it is approximately undersidpersed when $1/\nu < 1$ and approximately oversidpersed when $1/\nu > 1$. In this implementation, μ is excatly the mean, which differs from the COMPoissonReg package (Sellers & Lotze 2015).

The most commonly used GLM families (`binomial`, `poisson`) have fixed dispersion parameters which are internally ignored.

References

- Ferrari SLP, Cribari-Neto F (2004). "Beta Regression for Modelling Rates and Proportions." *J. Appl. Stat.* 31(7), 799-815.
- Hardin JW & Hilbe JM (2007). "Generalized linear models and extensions." Stata press.
- Sellers K & Lotze T (2015). "COMPoissonReg: Conway-Maxwell Poisson (COM-Poisson) Regression". R package version 0.3.5. https://CRAN.R-project.org/package=COMPoissonReg

simulate.glmmTMB *Simulate from a glmmTMB fitted model*

Description

Simulate from a glmmTMB fitted model

Usage

```
## S3 method for class 'glmmTMB'
simulate(object, nsim = 1, seed = NULL, ...)
```

Arguments

object	glmmTMB fitted model
nsim	number of response lists to simulate. Defaults to 1.
seed	random number seed
...	extra arguments

Details

Random effects are also simulated from their estimated distribution. Currently, it is not possible to condition on estimated random effects.

Value

returns a list of vectors. The list has length nsim. Each simulated vector of observations is the same size as the vector of response variables in the original data set. In the binomial family case each simulation is a two-column matrix with success/failure.

tmbroot *Compute likelihood profile confidence intervals of a TMB object by root-finding (generalized from TMB::tmbprofile)*

Description

Compute likelihood profile confidence intervals of a TMB object by root-finding (generalized from TMB::tmbprofile)

Usage

```
tmbroot(obj, name, target = 0.5 * qchisq(0.95, df = 1), lincomb,
  parm.range = c(NA, NA), sd.range = 7, trace = FALSE,
  continuation = FALSE)
```

Arguments

obj	a fitted glmmTMB object
name	parameter index/name
target	desired deviation from minimum log-likelihood. Default is set to retrieve the 95 if the objective function is a negative log-likelihood function
lincomb	linear combination of parameters
parm.range	lower and upper limits; if NA, a value will be guessed based on the parameter value and sd.range
sd.range	in the absence of explicit parm.range values, the range chosen will be the parameter value plus or minus sd.range. May be specified as a two-element vector for different ranges below and above the parameter value.
trace	report information?
continuation	use continuation method, i.e. set starting parameters for non-focal parameters to solutions from previous fits?

Value

a two-element numeric vector containing the lower and upper limits (or NA if the target is not achieved in the range), with an attribute giving the total number of function iterations used

---

vcov.glmmTMB *Calculate Variance-Covariance Matrix for a Fitted glmmTMB model*

Description

Calculate Variance-Covariance Matrix for a Fitted glmmTMB model

Usage

```
## S3 method for class 'glmmTMB'
vcov(object, full = FALSE, ...)
```

Arguments

object	a "glmmTMB" fit
full	return a full variance-covariance matrix?
...	ignored, for method compatibility

Value

By default (full==FALSE), a list of separate variance-covariance matrices for each model component (conditional, zero-inflation, dispersion). If full==TRUE, a single square variance-covariance matrix for *all* top-level model parameters (conditional, dispersion, and variance-covariance parameters)

Index