Package ‘actuaRE’

March 20, 2023

Type Package

Title Handling Hierarchically Structured Risk Factors using Random Effects Models

Version 0.1.5

Description Using this package, you can fit a random effects model using either the hierarchical credibility model, a combination of the hierarchical credibility model with a generalized linear model or a Tweedie generalized linear mixed model. See Campo, B.D.C. and Antonio, K. (2023) <doi:10.1080/03461238.2022.2161413>.

License GPL (>= 3)

Depends R (>= 3.5.0), stats, methods, cplm

Imports statmod, nlme, lme4, magrittr, data.table, ggplot2, knitr

Suggests plyr, insuranceData, actuar, utils, lattice, minqa

LazyData true

Encoding UTF-8

RoxygenNote 7.2.3

VignetteBuilder knitr

Config/testthat/edition 3

URL https://bavodc.github.io/websiteactuaRE/

NeedsCompilation no

Author Campo Bavo D.C. [aut, cre]

Maintainer Campo Bavo D.C. <bavo.decock@kuleuven.be>

Repository CRAN

Date/Publication 2023-03-20 20:50:02 UTC

R topics documented:

actuaRE-package .............................................................. 2
.addREs ............................................................................. 3
.adjustIntercept ................................................................. 4
**Description**

Using this package, you can fit a random effects model using either the hierarchical credibility model, a combination of the hierarchical credibility model with a generalized linear model or a Tweedie generalized linear mixed model. See Campo, B.D.C. and Antonio, K. (2023) <doi:10.1080/03461238.2022.2161413>.

**References**


burg: IIASA.  

sented at the Zurich ASTIN Colloquium*. [http://www.actuaries.org/ASTIN/Colloquia/Zurich/
Ohlsson.pdf](http://www.actuaries.org/ASTIN/Colloquia/Zurich/Ohlsson.pdf)


**See Also**

hierCredibility hierCredGLM hierCredTweedie tweedieGLMM BalanceProperty

**Examples**

```r
library(actuaRE)
# Vignette of the package
vignette(package = "actuaRE")

# Load data
data(hachemeisterLong)
data(dataCar)

# Hierarchical credibility model of Jewell
fit = hierCredibility(ratio, weight, cohort, state, hachemeisterLong)

# Combination of the hierarchical credibility model with a GLM (Ohlsson, 2008)
fit = hierCredGLM(Y ~ area + (1 | VehicleType / VehicleBody), dataCar, weights = w,
p = 1.7)
```

**.addREs**  
*Add random effects to the data frame*

**Description**

Internal function

**Usage**

```
.addREs(obj, newdata)
```

**Arguments**

- `obj` object with model fit
- `newdata` an object coercible to `data.table`.  

**Description**

This function updates the intercept term of the model fit such that the balance property is satisfied.

**Usage**

```r
adjustIntercept(obj, data)
```

**Arguments**

- `obj` an object of type `glm`, `cpglm` or `cpglm` containing the model fit.
- `data` a `data.frame` or `data.table` object that was used to fit the model.

**Value**

The object with the adjusted (fixed effects) coefficients.

**References**


**Examples**

```r
library(statmod)
dataCar = dataCar[1:1e3, ]
Fit = glm(Y ~ area + gender, data = datas, weights = datas$w, family = tweedie(1.75, 0),
model = TRUE, control = glm.control(epsilon = 1e-4, maxit = 5e2))
w = weights(Fit, "prior")
y = Fit$y
sum(w * y) == sum(w * fitted(Fit))
adjFit = adjustIntercept(Fit, datas)
coef(adjFit)
sum(w * y) == sum(w * fitted(adjFit))
```
**BalanceProperty**

---

**Description**

Function to assess whether the balance property holds

**Usage**

```r
BalanceProperty(obj)
```

**Arguments**

- `obj` an object containing the model fit

**Value**

a list with the slots `call` (the original call), `BalanceProperty` (logical indicating whether the balance property is satisfied) and `Alpha` (Ratio total observed damage to total predicted damage).

**References**


**Examples**

```r
fit = hierCredGLM(Y ~ area + (1 | VehicleType / VehicleBody), dataCar, weights = w, p = 1.75, epsilon = 1e-6)
BalanceProperty(fit)
```

---

**dataCar**

*data Car*

---

**Description**

This data set is taken from the `dataCar` data set of the `insuranceData` package and slightly adjusted (see the code in examples for reproducing this data set). The original data set is based on one-year vehicle insurance policies taken out in 2004 or 2005. There are 67566 policies, of which 4589 (6.8%) had at least one claim.
Usage

data(dataCar)

Format

A data frame with 67566 observations on the following 15 variables.

veh_value  vehicle value, in $10,000s
exposure  0-1
clm  occurrence of claim (0 = no, 1 = yes)
numclaims  number of claims
claimcst0  claim amount (0 if no claim)
veh_body  vehicle body, coded as BUS CONVT COUPE HBACK HDTOP MCARA MIBUS PANVN RDSTR SEDAN STNWG TRUCK UTE
veh_age  1 (youngest), 2, 3, 4
gender  a factor with levels F M
area  a factor with levels A B C D E F
agecat  1 (youngest), 2, 3, 4, 5, 6
X_OBSTAT_  a factor with levels 01101 0 0 0
Y  the loss ratio, defined as the number of claims divided by the exposure
w  the exposure, identical to exposure
VehicleType  type of vehicle, common vehicle or uncommon vehicle
VehicleBody  vehicle body, identical to veh_body

Details

Adjusted data set dataCar, where we removed claims with a loss ratio larger than 1 000 000. In addition, we summed the exposure per vehicle body and removed those where the summed exposure was less than 100. Hereby, we ensure that there is sufficient exposure for each vehicle body category.

Source

http://www.acst.mq.edu.au/GLMsforInsuranceData

References

Examples

```r
# How to construct the data set using the original dataCar data set from the insuranceData package
library(plyr)
library(magrittr)
data("dataCar", package = "insuranceData")
dataCar$Y = with(dataCar, claimcst0 / exposure)
dataCar$w = dataCar$exposure
dataCar = dataCar[which(dataCar$Y < 1e6),]
Yw = ddply(dataCar, .(veh_body), function(x) c(crossprod(x$Y, x$w) / sum(x$w), sum(x$w)))
dataCar = dataCar[!dataCar$veh_body %in% Yw[Yw$V2 < 1e2, "veh_body"],]
dataCar$veh_body %<>% droplevels()
dataCar$VehicleType = sapply(tolower(dataCar$veh_body), function(x) {
  if(x %in% c("sedan", "ute", "hback"))
    "Common vehicle"
  else
    "Uncommon vehicle"
})
dataCar$VehicleBody = dataCar$veh_body
```

findbars

Determine random-effects expressions from a formula

Description

From the right hand side of a formula for a mixed-effects model, determine the pairs of expressions that are separated by the vertical bar operator. Also expand the slash operator in grouping factor expressions and expand terms with the double vertical bar operator into separate, independent random effect terms.

Usage

```r
findbars(term)
```

Arguments

term  a mixed-model formula

Value

pairs of expressions that were separated by vertical bars

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: `mkRespMod`, `mkReTrms`, `nlformula`, `nobars`, `subbars`

Examples

```r
findbars(f1 <- Reaction ~ Days + (Days | Subject))
## => list( Days | Subject )
## These two are equivalent:% tests in ../inst/tests/test-doubleVertNotation.R
findbars(y ~ Days + (1 | Subject) + (0 + Days | Subject))
findbars(y ~ Days + (Days || Subject))
## => list of length 2: list ( 1 | Subject , 0 + Days | Subject)
findbars(~ 1 + (1 | batch / cask))
## => list of length 2: list ( 1 | cask:batch , 1 | batch)
```

---

**fixef**

*Extract fixed-effects estimates*

Description

Extract the fixed-effects estimates

Arguments

- **object**: any fitted model object from which fixed effects estimates can be extracted.

Details

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

Value

- a named, numeric vector of fixed-effects estimates.

Examples

```r
library(lme4)
fixef(lmer(Reaction ~ Days + (1|Subject) + (0+Days|Subject), sleepstudy))
fm2 <- lmer(Reaction ~ Days + Days2 + (1|Subject),
          data=transform(sleepstudy,Days2=Days))
fixef(fm2,add.dropped=TRUE)
## first two parameters are the same . . .
stopifnot(all.equal(fixef(fm2,add.dropped=TRUE)[1:2],
                    fixef(fm2)))
```
fixef-actuaRE

Extract the fixed-effects estimates from a fitted random effects model

Description

A generic function to extract the fixed effects (i.e. the company-specific effects) estimates from a fitted random effects model.

Usage

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

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

Arguments

object an object of type `hierCredGLM` or `hierCredTweedie`

... ignored.

Value

a named, numeric vector of fixed-effects estimates.

Examples

```r
fit = hierCredGLM(Y ~ area + (1 | VehicleType / VehicleBody), dataCar, weights = w, p = 1.75, epsilon = 1e-6)
fixef(fit)
```

hachemeisterLong

Hachemeister Data Set

Description

Long format of the Hachemeister (1975) data set giving average claim amounts in private passenger bodily injury insurance. We have data of five U.S. states over 12 quarters between July 1970 and June 1973 and we have the corresponding number of claims. To obtain a hierarchical structure, we created an artificial variable cohort. With this, we created a hierarchical multi-level factor, with cohort as the first hierarchical level and state as the second hierarchical level, nested within cohort.
Usage

hachemeisterLong

Format

A data.frame with 60 rows and the following 5 columns:

- cohort: artificially created variable;
- state: the state number;
- time: time variable (quarter of the observation);
- ratio: the average claim amount;
- weight: the corresponding number of claims.

Source


hierCredGLM

Combining the hierarchical credibility model with a GLM (Ohlsson, 2008)

Description

Fit a random effects model using Ohlsson’s methodology. In this function you explicitly specify the power parameter p. See hierCredTweedie when you also want to estimate the p.

Usage

hierCredGLM(
  formula,  # formula
  data,     # data
  weights,  # weights
  p = 1.5,   # power parameter
  link.power = 0,  # link power
  muHatGLM = TRUE,  # estimate mu
  epsilon = 1e-04,  # tolerance
  maxiter = 500,    # maximum number of iterations
  maxiterGLM = 500, # maximum number of iterations for GLM
  verbose = FALSE,  # verbose output
  returnData = TRUE,  # return data
  balanceProperty = TRUE,  # balance property
  y = TRUE,
  ...  # additional arguments
)
Arguments

- **formula**: object of type `formula` that specifies which model should be fitted. Syntax is the same as for `lmer` and `glmer`. For example, `Yijkt ~ x1 + x2 + (1 | Industry / Branch)`.

- **data**: an object that is coercible by `as.data.table`, containing the variables in the model.

- **weights**: variable name of the exposure weight.

- **p**: the value for the power parameter of the Tweedie distribution, which is passed to `tweedie`. Default is 1.5.

- **link.power**: index of power link function, which is passed to `tweedie`. `link.power = 0` produces a log-link. Defaults to the canonical link, which is $1 - p$.

- **muHatGLM**: indicates which estimate has to be used in the algorithm for the intercept term. Default is `TRUE`, which used the intercept as estimated by the GLM. If `FALSE`, the estimate of the hierarchical credibility model is used.

- **epsilon**: positive convergence tolerance $\epsilon$; the iterations converge when $\frac{||\theta[k] - \theta[k-1]||^2[2]}{||\theta[k] - \theta[k-1]||^2[2]} < \epsilon$. Here, $\theta[k]$ is the parameter vector at the $k^{th}$ iteration.

- **maxiter**: maximum number of iterations.

- **maxiterGLM**: maximum number of iterations when fitting the GLM part. Passed to `glm`.

- **verbose**: logical indicating if output should be produced during the algorithm.

- **returnData**: logical indicating if input data has to be returned.

- **balanceProperty**: logical indicating if the balance property should be satisfied.

- **y**: logical indicating whether the response vector should be returned as a component of the returned value.

- **...**: arguments passed to `glm`

Value

An object of type `hierCredGLM` with the following slots:

- **call**: the matched call

- **HierarchicalResults**: results of the hierarchical credibility model.

- **fitGLM**: the results from fitting the GLM part.

- **iter**: total number of iterations.

- **Converged**: logical indicating whether the algorithm converged.

- **LevelsCov**: object that summarizes the unique levels of each of the contract-specific covariates.

- **fitted.values**: the fitted mean values, resulting from the model fit.

- **prior.weights**: the weights (exposure) initially supplied.

- **y**: if requested, the response vector. Default is `TRUE`. 

---

`hierCredGLM`
References


See Also

hierCredGLM-class, fitted.hierCredGLM, predict.hierCredGLM, ranef.actuaRE, weights.actuaRE, hierCredibility, hierCredTweedie, plotRE, adjustIntercept, BalanceProperty

Examples

data("dataCar")
fit = hierCredGLM(Y ~ area + (1 | VehicleType / VehicleBody), dataCar, weights = w, p = 1.7)
fit
summary(fit)
ranef(fit)
fixef(fit)

hierCredGLM-class

Class "hierCredGLM" of fitted random effects models estimated with Ohlsson's GLMC algorithm

Description

Class "hierCredGLM" of fitted random effects models estimated with Ohlsson’s GLMC algorithm

Usage

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

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

## S3 method for class 'hierCredGLM'
fitted(object, ...)

Arguments

x       an object of class hierCredGLM
...      currently ignored.
object   an object of class hierCredGLM
Value

The function `hierCredGLM` returns an object of class `hierCredGLM`, which has the following slots:

- **call**: the matched call
- **HierarchicalResults**: results of the hierarchical credibility model.
- **fitGLM**: the results from fitting the GLM part.
- **iter**: total number of iterations.
- **Converged**: logical indicating whether the algorithm converged.
- **LevelsCov**: object that summarizes the unique levels of each of the contract-specific covariates.
- **fitted.values**: the fitted mean values, resulting from the model fit.
- **prior.weights**: the weights (exposure) initially supplied.
- **y**: if requested, the response vector. Default is TRUE.

S3 methods

- **print**: Prints the call, the estimated variance parameters, the unique number of categories of the hierarchical MLF and the output of the GLM part. The ... argument is currently ignored. Returns an invisible copy of the original object.
- **summary**: In addition to the output of the print.hierCredGLM function, the summary function also prints the random effect estimates and a summary of the GLM (see `summary.glm`). Returns an invisible copy of the original object.
- **fitted**: Returns the fitted values.

See Also

- `hierCredGLM`

Description

Fit a random effects model, without contract-specific risk factors, using the hierarchical credibility model of Jewell.
Usage

\[
\text{hierCredibility}(\ Yijkt, \ wijkt, \ sector, \ group, \ data, \ muHat = \text{NULL}, \ type = \text{c("additive", "multiplicative")}, \ returnData = \text{FALSE})
\]

Arguments

- **Yijkt**: variable name of the response variable (the loss cost within actuarial applications).
- **wijkt**: variable name of the exposure weight.
- **sector**: variable name of the first hierarchical level.
- **group**: variable name of the second hierarchical level that is nested within the first hierarchical level.
- **data**: an object that is coercible by `as.data.table`, containing the variables in the model.
- **muHat**: estimate for the intercept term. Default is `NULL` and in this case, the estimator as given in Ohlsson (2005) is used.
- **type**: specifies whether the additive (Dannenburg, 1996) or multiplicative (Ohlsson, 2005) formulation of the hierarchical credibility model is used. Default is additive.
- **returnData**: Logical, indicates whether the data object has to be returned. Default is `FALSE`.

Value

An object of type `hierCredibility` with the following slots:

- **call**: the matched call.
- **type**: Whether additive or multiplicative hierarchical credibility model is used.
- **Variances**: The estimated variance components. \(s^2\) is the estimated variance of the individual contracts, \(\tau_{\text{as}}\) the estimate of \(\text{Var}(V[j])\) and \(\nu_{\text{as}}\) is the estimate of \(\text{Var}(V[jk])\).
- **Means**: The estimated averages at the portfolio level (intercept term \(\mu\)), at the first hierarchical level \(\text{bar}(Y)[\%\%j\%\%\%\%]\) and at the second hierarchical level \(\text{bar}(Y)[\%\%j\%k\%\%]\).
- **Weights**: The weights at the first hierarchical level \(w[j\%\%]\) and at the second hierarchical level \(w[jk\%\%]\).
- **Credibility**: The credibility weights at the first hierarchical level \(q[j\%\%]\) and at the second hierarchical level \(z[jk]\).
### Premiums
The overall expectation $\widehat{\mu}$, sector expectation $\widehat{V}_j$ and group expectation $\widehat{V}_{jk}$.

### Relativity
The estimated random effects $\widehat{U}_j$ and $\widehat{U}_{jk}$ of the sector and group, respectively.

### RawResults
Objects of type `data.table` with all intermediate results.

### fitted.values
The fitted mean values, resulting from the model fit.

### References

### See Also
- `hierCredibility-class`, `fitted.hierCredibility`, `predict.hierCredibility`, `ranef-actuaRE`, `weights-actuaRE`, `hierCredTweedie`, `hierCredGLM`, `cpglm`, `plotRE`

### Examples
```r
library(actuar)
library(actuaRE)
data("hachemeister", package = "actuar")
Df = as.data.frame(hachemeister)
X = as.data.frame(cbind(cohort = c(1, 2, 1, 2, 2), hachemeister))
Df = reshape(X, idvar = "state", varying = list(paste0("ratio.", 1:12),
paste0("weight.", 1:12)), direction = "long")
fitActuar = cm(~ cohort + cohort:state, data = X, ratios = ratio.1:ratio.12,
weights = weight.1:weight.12, method = "Ohlsson")
fitActuaRE = hierCredibility(ratio.1, weight.1, cohort, state, Df)
summary(fitActuar)
summary(fitActuaRE)
```

---

**hierCredibility-class**

Class "hierCredibility" of fitted hierarchical credibility models

**Description**

Class "hierCredibility" of fitted hierarchical credibility models
Usage

```r
## S3 method for class 'hierCredibility'
print(x, ...)

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

## S3 method for class 'hierCredibility'
fitted(object, ...)
```

Arguments

- **x**: an object of class `hierCredibility`
- **...**: currently ignored.
- **object**: an object of class `hierCredibility`

Value

The function `hierCredibility` returns an object of class `hierCredibility`, which has the following slots:

- **call**: the matched call
- **type**: Whether additive or multiplicative hierarchical credibility model is used.
- **Variances**: The estimated variance components. `s2` is the estimated variance of the individual contracts, `tausq` the estimate of $\text{Var}(V[j])$ and `nusq` is the estimate of $\text{Var}(V[jk])$.
- **Means**: The estimated averages at the portfolio level (intercept term $\mu$), at the first hierarchical level ($\bar{Y}[j]$, $\bar{Y}[jk]$) and at the second hierarchical level ($\bar{Y}[%.jk%.]$).
- **Weights**: The weights at the first hierarchical level $w[j%.jk%.]$ and at the second hierarchical level $z[j%.jk%.]$.
- **Credibility**: The credibility weights at the first hierarchical level $q[j%.jk%.]$ and at the second hierarchical level $z[jk]$.  
- **Premiums**: The overall expectation $\hat{\mu}$, sector expectation $\hat{\mu}[j]$ and group expectation $\hat{\mu}[jk]$. 
- **Relativity**: The estimated random effects $\hat{U}[j]$ and $\hat{U}[jk]$ of the sector and group, respectively.
- **RawResults**: Objects of type `data.table` with all intermediate results.
- **fitted.values**: the fitted mean values, resulting from the model fit.

S3 methods

- **print**: Prints the call, the estimated variance parameters and the unique number of categories of the hierarchical MLF. The ... argument is currently ignored. Returns an invisible copy of the original object.
summary: In addition to the output of the `print.hierCredibility` function, the `summary` function prints the random effect estimates as well. Returns an invisible copy of the original object.

fitted: Returns the fitted values.

See Also

`hierCredibility`

---

### hierCredTweedie

*Combining the hierarchical credibility model with a GLM (Ohlsson, 2008)*

#### Description

Fit a random effects model using Ohlsson’s methodology. In this function you estimate the power parameter p. See `hierCredGLM` when you want fix p.

#### Usage

```r
hierCredTweedie(
  formula,
  data,
  weights,
  muHatGLM = TRUE,
  epsilon = 1e-04,
  maxiter = 500,
  verbose = FALSE,
  returnData = TRUE,
  cpglmControl = list(bound.p = c(1.01, 1.99)),
  balanceProperty = TRUE,
  optimizer = "bobyqa",
  y = TRUE,
  ...
)
```

#### Arguments

- `formula`: object of type `formula` that specifies which model should be fitted. Syntax is the same as for `lmer` and `glmer`. For example, \( Y_{ijkt} \sim x_1 + x_2 + (1 | \text{Industry} / \text{Branch}) \).
- `data`: an object that is coercible by `as.data.table`, containing the variables in the model.
- `weights`: variable name of the exposure weight.
- `muHatGLM`: indicates which estimate has to be used in the algorithm for the intercept term. Default is `TRUE`, which used the intercept as estimated by the GLM. If `FALSE`, the estimate of the hierarchical credibility model is used.
epsilon: positive convergence tolerance $\epsilon$; the iterations converge when $||\mathbf{\theta}[k] - \mathbf{\theta}[k-1]||^2 ||\mathbf{\theta}[k-1]||^2 < \epsilon$. Here, $\mathbf{\theta}[k]$ is the parameter vector at the $k^{th}$ iteration.

maxiter: maximum number of iterations.

verbose: logical indicating if output should be produced during the algorithm.

returnData: logical indicating if input data has to be returned.

cpglmControl: a list of parameters to control the fitting process in the GLM part. By default, `cpglmControl = list(bound.p = c(1.01, 1.99))` which restricts the range of the power parameter $p$ to $[1.01, 1.99]$ in the fitting process. This list is passed to `cpglm`.

balanceProperty: logical indicating if the balance property should be satisfied.

optimizer: a character string that determines which optimization routine is to be used in estimating the index and the dispersion parameters. Possible choices are "nlminb" (the default, see `nlminb`), "bobyqa" (bobyqa) and "L-BFGS-B" (optim).

y: logical indicating whether the response vector should be returned as a component of the returned value.

...: arguments passed to `cpglm`.

Details

When estimating the GLM part, this function uses the `cpglm` function from the `cplm` package.

Value

An object of type `hierCredTweedie` with the following slots:

call: the matched call
HierarchicalResults: results of the hierarchical credibility model.
fitGLM: the results from fitting the GLM part.
iter: total number of iterations.
Converged: logical indicating whether the algorithm converged.
LevelsCov: object that summarizes the unique levels of each of the contract-specific covariates.
fitted.values: the fitted mean values, resulting from the model fit.
prior.weights: the weights (exposure) initially supplied.
y: if requested, the response vector. Default is TRUE.

References

See Also

hierCredTweedie-class, fitted.hierCredTweedie, predict.hierCredTweedie, ranef-actuaRE, weights-actuaRE, hierCredibility, hierCredGLM, cpglm, plotRE, adjustIntercept, BalanceProperty


Examples

```r
data("dataCar")
fit = hierCredTweedie(Y ~ area + (1 | VehicleType / VehicleBody), dataCar, weights = w, epsilon = 1e-6)
fit
summary(fit)
ranef(fit)
fixef(fit)
```

hierCredTweedie-class

Class “hierCredTweedie” of fitted random effects models estimated with Ohlsson’s GLMC algorithm

Description

Class "hierCredTweedie" of fitted random effects models estimated with Ohlsson’s GLMC algorithm

Usage

```r
## S3 method for class 'hierCredTweedie'
print(x, ...)

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

## S3 method for class 'hierCredTweedie'
fitted(object, ...)
```

Arguments

- `x` an object of class `hierCredTweedie`
- `...` currently ignored.
- `object` an object of class `hierCredTweedie`
Value
The function `hierCredGLM` returns an object of class `hierCredGLM`, which has the following slots:

- **call**: the matched call
- **HierarchicalResults**: results of the hierarchical credibility model.
- **fitGLM**: the results from fitting the GLM part.
- **iter**: total number of iterations.
- **Converged**: logical indicating whether the algorithm converged.
- **LevelsCov**: object that summarizes the unique levels of each of the contract-specific covariates.
- **fitted.values**: the fitted mean values, resulting from the model fit.
- **prior.weights**: the weights (exposure) initially supplied.
- **y**: if requested, the response vector. Default is `TRUE`.

S3 methods
- **print**: Prints the call, the estimated variance parameters, the unique number of categories of the hierarchical MLF and the output of the GLM part. The `...` argument is currently ignored. Returns an invisible copy of the original object.
- **summary**: In addition to the output of the `print.hierCredTweedie` function, the `summary` function also prints the random effect estimates and a summary of the GLM (see `summary.glm`). Returns an invisible copy of the original object.
- **fitted**: Returns the fitted values.

See Also
`hierCredTweedie`

### is.formula

<table>
<thead>
<tr>
<th>is.formula</th>
<th>Formula</th>
</tr>
</thead>
</table>

Description
Checks if the object is of the type formula

Usage
`is.formula(x)`

Arguments

- **x**: the object.
Value

logical indicating if the object is a formula or not.

Examples

\[
f = \text{formula}(y \sim x) \\
is.\text{formula}(f)
\]

\[
\begin{array}{ll}
\text{isNested} & \text{Is } f1 \text{ nested within } f2? \\
\end{array}
\]

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

\[
is.\text{Nested}(f1, f2)
\]

Arguments

- \( f1 \): factor 1
- \( f2 \): factor 2

Value

TRUE if factor 1 is nested within factor 2

Examples

\[
\text{library(lme4)} \\
\text{with(Pastes, isNested(cask, batch)) } ## \Rightarrow \text{FALSE} \\
\text{with(Pastes, isNested(sample, batch)) } ## \Rightarrow \text{TRUE}
\]
Modular Functions for Mixed Model Fits

Description

Modular functions for mixed model fits

Usage

```r
glFormula(formula, data = NULL, family = gaussian,
          subset, weights, na.action, offset, contrasts = NULL,
          start, mustart, etastart, control = glmerControl(), ...)
```

Arguments

- **formula**: a two-sided linear formula object describing both the fixed-effects and random-effects parts of the model, with the response on the left of a `~` operator and the terms, separated by `+` operators, on the right. Random-effects terms are distinguished by vertical bars (`|`) separating expressions for design matrices from grouping factors.
- **data**: an optional data frame containing the variables named in `formula`. By default the variables are taken from the environment from which `lmer` is called. While `data` is optional, the package authors strongly recommend its use, especially when later applying methods such as `update` and `drop1` to the fitted model. If `data` is omitted, variables will be taken from the environment of `formula` (if specified as a formula) or from the parent frame (if specified as a character vector).
- **subset**: an optional expression indicating the subset of the rows of `data` that should be used in the fit. This can be a logical vector, or a numeric vector indicating which observation numbers are to be included, or a character vector of the row names to be included. All observations are included by default.
- **weights**: an optional vector of ‘prior weights’ to be used in the fitting process. Should be `NULL` or a numeric vector.
- **na.action**: a function that indicates what should happen when the data contain NAs. The default action (`na.omit`, inherited from the ‘factory fresh’ value of `getOption("na.action")`) strips any observations with any missing values in any variables.
- **offset**: this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be `NULL` or a numeric vector of length equal to the number of cases. One or more `offset` terms can be included in the formula instead or as well, and if more than one is specified their sum is used. See `model.offset`.
- **contrasts**: an optional `list`. See the `contrasts.arg` of `model.matrix.default`.
- **control**: a list giving
  ```r
  for [gl]Formula: all options for running the model, see `lmerControl`;`
for mkLmerDevfun, mkGlmerDevfun: options for the inner optimization step;
for optimizeLmer and optimizeGlmer: control parameters for nonlinear optimizer (typically inherited from the . . . argument to lmerControl).

start
starting values (see lmer; for glFormula, should be just a numeric vector of fixed-effect coefficients)

family
a GLM family; see glm and family.

mustart
optional starting values on the scale of the conditional mean; see glm for details.

etastart
optional starting values on the scale of the unbounded predictor; see glm for details.

... other potential arguments; for optimizeLmer and optimizeGlmer, these are passed to internal function optwrap, which has relevant parameters calc.derivs and use.last.params (see lmerControl).

Details

These functions make up the internal components of an [gn]lmer fit.

- [g]lFormula takes the arguments that would normally be passed to [g]lmer, checking for errors and processing the formula and data input to create a list of objects required to fit a mixed model.
- mk(Gl|L)merDevfun takes the output of the previous step (minus the formula component) and creates a deviance function
- optimize(Gl|L)mer takes a deviance function and optimizes over theta (or over theta and beta, if stage is set to 2 for optimizeGlmer
- updateGlmerDevfun takes the first stage of a GLMM optimization (with nAGQ=0, optimizing over theta only) and produces a second-stage deviance function
- mkMerMod takes the environment of a deviance function, the results of an optimization, a list of random-effect terms, a model frame, and a model all and produces a [g]lmerMod object.

Value

lFormula and glFormula return a list containing components:

fr model frame
X fixed-effect design matrix
reTrms list containing information on random effects structure: result of mkReTrms

mkLmerDevfun and mkGlmerDevfun return a function to calculate deviance (or restricted deviance) as a function of the theta (random-effect) parameters. updateGlmerDevfun returns a function to calculate the deviance as a function of a concatenation of theta and beta (fixed-effect) parameters. These deviance functions have an environment containing objects required for their evaluation. CAUTION: The environment of functions returned by mk(Gl|L)merDevfun contains reference class objects (see ReferenceClasses, merPredD-class, lmResp-class), which behave in ways that may surprise many users. For example, if the output of mk(Gl|L)merDevfun is naively copied, then modifications to the original will also appear in the copy (and vice versa). To avoid this behavior one must make a deep copy (see ReferenceClasses for details).

optimizeLmer and optimizeGlmer return the results of an optimization.
Examples

library(lme4)
### Fitting a linear mixed model in 4 modularized steps

### 1. Parse the data and formula:
lmod <- lFormula(Reaction ~ Days + (Days|Subject), sleepstudy)
names(lmod)
### 2. Create the deviance function to be optimized:
(devfun <- do.call(mkLmerDevfun, lmod))
ls(environment(devfun)) # the environment of 'devfun' contains objects
# required for its evaluation
### 3. Optimize the deviance function:
opt <- optimizeLmer(devfun)
opt[1:3]
### 4. Package up the results:
mkMerMod(environment(devfun), opt, lmod$reTrms, fr = lmod$fr)

### Same model in one line
lmer(Reaction ~ Days + (Days|Subject), sleepstudy)

### Fitting a generalized linear mixed model in six modularized steps

### 1. Parse the data and formula:
glmod <- glFormula(cbind(incidence, size - incidence) ~ period + (1 | herd),
                    data = cbpp, family = binomial)
#.... see what've got :
str(glmod, max=1, give.attr=FALSE)
### 2. Create the deviance function for optimizing over theta:
(devfun <- do.call(mkGlmerDevfun, glmod))
ls(environment(devfun)) # the environment of devfun contains lots of info
### 3. Optimize over theta using a rough approximation (i.e. nAGQ = 0):
(opt <- optimizeGlmer(devfun))
### 4. Update the deviance function for optimizing over theta and beta:
(devfun <- updateGlmerDevfun(devfun, glmod$reTrms))
### 5. Optimize over theta and beta:
opt <- optimizeGlmer(devfun, stage=2)
str(opt, max=1) # seeing what we 'got'
### 6. Package up the results:
(fMod <- mkMerMod(environment(devfun), opt, glmod$reTrms, fr = glmod$fr))

### Same model in one line
fM <- glmer(cbind(incidence, size - incidence) ~ period + (1 | herd),
            data = cbpp, family = binomial)
all.equal(fMod, fM, check.attributes=FALSE, tolerance = 1e-12)
# ---- -- even tolerance = 0 may work
**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

```r
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: `findbars`, `mkRespMod`, `mkReTrms`, `nlformula`, `subbars`

Examples

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

---

**NrUnique**

Number of unique elements in a vector

Description

Number of unique elements in a vector

Usage

```r
NrUnique(x, na.rm = TRUE)
```
Arguments

- x: object of type vector.
- na.rm: logical indicating if missing values have to be removed. Default to TRUE.

Value

vector with the number of unique elements

Examples

```r
set.seed(1)
x = sample(letters, 50, TRUE)
NrUnique(x)
```

plotRE

Visualizing the random effect estimates using ggplot2

Description

Using this function, you can create plots of the random effect estimates from fitted random effects models. To make the plots, we rely on the `ggplot2` package.

Usage

```r
plotRE(
  obj,
  levelRE = c("all", "first", "second"),
  colour = "black",
  plot = TRUE
)
```

Arguments

- obj: an object of type `hierCredibility`, `hierCredGLM` or `hierCredTweedie`
- levelRE: indicates which hierarchical level has to be used. "all" plots both levels in the hierarchy, "first" the first level in the hierarchy and "second" the second level.
- colour: colour for `geom_point`
- plot: logical indicating if the `ggplot` objects have to be plotted.

Value

a list with `ggplot` objects.
predict.hierCredGLM

Examples

```r
fitHGLM <- hierCredGLM(Y ~ area + gender + (1 | VehicleType / VehicleBody), dataCar, weights = w)
plotRE(fitHGLM)
```

predict.hierCredGLM  Model predictions

Description

Obtain predictions based on the model fit with `hierCredGLM`

Usage

```r
## S3 method for class 'hierCredGLM'
predict(object, newdata = NULL, ...)
```

Arguments

- `object` a model object for which prediction is desired.
- `newdata` optionally, a data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
- `...` arguments passed to `glm`

Details

The random effects are taken into account by specifying these as an offset in the `predict.glm` function.

Value

If `newdata` is omitted the predictions are based on the data used for the fit.

See Also

`hierCredGLM`
predict.hierCredibility

Model predictions

Description

Obtain predictions based on a model fit with hierCredibility

Usage

## S3 method for class 'hierCredibility'
predict(object, newdata = NULL, ...)

Arguments

- **object**: a model object for which prediction is desired.
- **newdata**: optionally, a data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
- **...**: ignored.

Value

If newdata is omitted the predictions are based on the data used for the fit.

See Also

hierCredibility

predict.hierCredTweedie

Model predictions

Description

Obtain predictions based on the model fit with hierCredTweedie

Usage

## S3 method for class 'hierCredTweedie'
predict(object, newdata = NULL, ...)

Arguments

- **object**: a model object for which prediction is desired.
- **newdata**: optionally, a data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
- **...**: arguments passed to cpglm
Details

The random effects are taken into account by specifying these as an offset in the predict.cpglm function.

Value

If newdata is omitted the predictions are based on the data used for the fit.

See Also

hierCredTweedie
_ranef_  
\textit{Extract the modes of the random effects}

**Description**

A generic function to extract the conditional modes of the random effects from a fitted model object. For linear mixed models the conditional modes of the random effects are also the conditional means.

**Arguments**

- **object**
  
an object of a class of fitted models with random effects.

**Details**

If grouping factor i has k levels and j random effects per level the ith component of the list returned by \texttt{ranef} is a data frame with k rows and j columns.

**Value**

- From \texttt{ranef}: An object composed of a list of data frames, one for each grouping factor for the random effects. The number of rows in the data frame is the number of levels of the grouping factor. The number of columns is the dimension of the random effect associated with each level of the factor.

**Examples**

```r
library(lattice) # for dotplot, qqmath
library(lme4)
fml <- lmer(Reaction ~ Days + (Days|Subject), sleepstudy)
fml2 <- lmer(Reaction ~ Days + (1|Subject) + (0+Days|Subject), sleepstudy)
fml3 <- lmer(diameter ~ (1|plate) + (1|sample), Penicillin)
ranef(fml1)
```

---

_\texttt{ranef-actuaRE}  
Extract the random effect estimates from a fitted random effects model_

**Description**

A generic function to extract the estimates/predictions of the random effects from a fitted random effects model.
tweedieGLMM  
Fitting a Tweedie GLMM, using the initial estimates of hierCredTweedie

Usage

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

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

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

Arguments

- `object`: an object of type `hierCredibility`, `hierCredGLM` or `hierCredTweedie`
- `...`: Currently ignored.

Value

A list of data frames, one for each grouping factor for the random effects. The number of rows in the data frame is the number of levels of the grouping factor. The first (two) columns correspond(s) to the grouping factor. The last column corresponds to the estimated random effect.

Description

This function first estimates the random effects model using Ohlsson’s GLMC algorithm (Ohlsson, 2008) and then uses these estimates as initial estimates when fitting a Tweedie GLMM.

Usage

```r
tweedieGLMM(
    formula,
    data,
    weights,
    muHatGLM = FALSE,
    epsilon = 1e-04,
    maxiter = 500,
    verbose = FALSE,
    balanceProperty = TRUE
)
```
Arguments

- **formula**: object of type `formula` that specifies which model should be fitted. Syntax is the same as for `lmer` and `glmer`. For example, `Yijkt ~ x1 + x2 + (1 | Industry / Branch).

- **data**: an object that is coercible by `as.data.table`, containing the variables in the model.

- **weights**: variable name of the exposure weight.

- **muHatGLM**: indicates which estimate has to be used in the algorithm for the intercept term. Default is `TRUE`, which used the intercept as estimated by the GLM. If `FALSE`, the estimate of the hierarchical credibility model is used.

- **epsilon**: positive convergence tolerance $\epsilon$; the iterations converge when $||\theta[k] - \theta[k-1]||^2 < \epsilon$. Here, $\theta[k]$ is the parameter vector at the $k^{th}$ iteration.

- **maxiter**: maximum number of iterations.

- **verbose**: logical indicating if output should be produced during the algorithm.

- **balanceProperty**: logical indicating if output should be produced during the algorithm.

Value

an object of class `cpglmm`, containing the model fit.

References


See Also

- `cpglmm` and `hierCredTweedie`

Examples

```r
data("dataCar")
fitTweedieGLMM = tweedieGLMM(Y ~ area + gender + (1 | VehicleType / VehicleBody), dataCar, weights = w, verbose = TRUE, epsilon = 1e-4)
```
weights-actuarE

Extract the model weights

Description

weights is a generic function which extracts fitting weights from objects returned by modeling functions. Methods can make use of napredict methods to compensate for the omission of missing values. The default methods does so.

Usage

```r
## S3 method for class 'cpglm'
weights(object, type = c("prior", "working"), ...)

## S3 method for class 'hierCredGLM'
weights(object, type = c("prior", "working"), ...)

## S3 method for class 'hierCredTweedie'
weights(object, type = c("prior", "working"), ...)
```

Arguments

- `object`: an object for which the extraction of model weights is meaningful. Can be either `cpglm`, `glm`, `hierCredibility`, `hierCredGLM` or `hierCredTweedie`
- `type`: indicates if prior or working weights need to be extracted.
- `...`: ignored

Value

Weights extracted from the object object: the default method looks for component "weights" and if not NULL calls napredict on it.

See Also

`weights`, `cpglm`, `glm`, `hierCredibility`, `hierCredGLM` or `hierCredTweedie`
Index

* datasets
  dataCar, 5
  hachemeisterLong, 9
* methods
  ranef, 30
* models
  findbars, 7
  fixef, 8
  modular, 22
  nobars, 25
  ranef, 30
* package
  actuaRE-package, 2
* utilities
  findbars, 7
  nobars, 25
  .addREs, 3

actuaRE (actuaRE-package), 2
actuaRE-package, 2
adjustIntercept, 4, 12, 19
as.data.table, 11, 14, 17, 32
BalanceProperty, 3, 5, 12, 19, 29
bobyqa, 18
cpglm, 4, 15, 18, 19, 28, 33
cpglmm, 4, 32
data.frame, 4
data.table, 4
dataCar, 5, 5
environment, 23
family, 23
findbars, 7, 25
fitted.hierCredGLM, 12
fitted.hierCredGLM (hierCredGLM-class), 12
fitted.hierCredibility, 15
fitted.hierCredibility (hierCredibility-class), 15
fitted.hierCredTweedie, 19
fitted.hierCredTweedie (hierCredTweedie-class), 19
fixed.effects (fixef), 8
fixef, 8
fixef-actuaRE, 9
fixef.hierCredGLM (fixef-actuaRE), 9
fixef.hierCredTweedie (fixef-actuaRE), 9
formula, 8, 11, 17, 25, 32
geom_point, 26
ggplot, 26
ggplot2, 26
glFormula (modular), 22
glm, 4, 23, 33
glmer, 11, 17, 32
hachemeisterLong, 9
hierCredGLM, 3, 9, 10, 12, 13, 15, 19, 20, 26, 27, 31, 33
hierCredGLM-class, 12
hierCredibility, 3, 12, 13, 16, 17, 19, 26, 28, 31, 33
hierCredibility-class, 15
hierCredTweedie, 3, 9, 10, 12, 15, 17, 19, 20, 26, 29, 31–33
hierCredTweedie-class, 19
is.formula, 20
isNested, 21
list, 22
lmer, 11, 17, 23, 32
lmerControl, 22, 23
mkMerMod, 23
mkRespMod, 8, 25
mkReTrms, 8, 23, 25
model.frame, 8, 25
INDEX

model.matrix, 8, 25
model.matrix.default, 22
model.offset, 22
modular, 22

napredict, 33
nlformula, 8, 25
nlminb, 18
nobars, 8, 25
NrUnique, 25

offset, 22
optim, 18

plotRE, 12, 15, 19, 26
predict.hierCredGLM, 12, 27
predict.hierCredibility, 15, 28
predict.hierCredTweedie, 19, 28
print.BalanceProperty, 29
print.hierCredGLM (hierCredGLM-class), 12
print.hierCredibility (hierCredibility-class), 15
print.hierCredTweedie (hierCredTweedie-class), 19

ranef, 30
ranef-actuaRE, 30
ranef.hierCredGLM (ranef-actuaRE), 30
ranef.hierCredibility (ranef-actuaRE), 30
ranef.hierCredTweedie (ranef-actuaRE), 30
ReferenceClasses, 23

subbars, 8, 25
summary.glm, 13, 20
summary.hierCredGLM
  (hierCredGLM-class), 12
summary.hierCredibility
  (hierCredibility-class), 15
summary.hierCredTweedie
  (hierCredTweedie-class), 19

tweedie, 11
tweedieGLMM, 3, 31

weights, 33
weights-actuaRE, 33
weights.cpglm (weights-actuaRE), 33