Package 'actuaRE'

December 2, 2025

Type Package

Title Handling Hierarchically Structured Risk Factors using Random Effects Models

Version 0.1.7

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, reformulas

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

LazyData true

Encoding UTF-8

RoxygenNote 7.3.1

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 2025-12-02 11:10:02 UTC

Contents

`actuaRE-package` *Handling Hierarchically Structured Risk Factors using Random Effects Models*

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

Campo, B.D.C. and Antonio, Katrien (2023). Insurance pricing with hierarchically structured data an illustration with a workers' compensation insurance portfolio. *Scandinavian Actuarial Journal*, doi: 10.1080/03461238.2022.2161413

Dannenburg, D. R., Kaas, R. and Goovaerts, M. J. (1996). *Practical actuarial credibility models*. Amsterdam: IAE (Institute of Actuarial Science and Econometrics of the University of Amsterdam).

Jewell, W. S. (1975). *The use of collateral data in credibility theory: a hierarchical model*. Laxenburg: IIASA.

Ohlsson, E. (2005). Simplified estimation of structure parameters in hierarchical credibility. *Presented at the Zurich ASTIN Colloquium*.

Ohlsson, E. (2008). Combining generalized linear models and credibility models in practice. *Scandinavian Actuarial Journal* **2008**(4), 301–314.

See Also

`hierCredibility` `hierCredGLM` `hierCredTweedie` `tweedieGLMM` `BalanceProperty`

Examples

```
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.

---

adjustIntercept *Adjust the intercept to regain the balance property*

Description

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

Usage

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

Arguments

obj	an object of type glm, cpglm or cpglmm 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

Campo, B.D.C. and Antonio, Katrien (2023). Insurance pricing with hierarchically structured data an illustration with a workers' compensation insurance portfolio. *Scandinavian Actuarial Journal*, doi: 10.1080/03461238.2022.2161413

Wüthrich, M. V. (2020). Bias regularization in neural network models for general insurance pricing. *European actuarial journal* **10**(1), 179–202.

Examples

```
library(statmod)
datas  = 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 *Balance property*

Description

Function to assess whether the balance property holds

Usage

```
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

Campo, B.D.C. and Antonio, Katrien (2023). Insurance pricing with hierarchically structured data an illustration with a workers' compensation insurance portfolio. *Scandinavian Actuarial Journal*, doi: 10.1080/03461238.2022.2161413

Wüthrich, M. V. (2020). Bias regularization in neural network models for general insurance pricing. *European actuarial journal* **10**(1), 179–202.

Examples

```
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

De Jong P., Heller G.Z. (2008), Generalized linear models for insurance data, Cambridge University Press

Examples

```
# 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

```
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

```
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

```
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

```
## 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

```
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

Hachemeister, C. A. (1975), *Credibility for regression models with application to trend*, Proceedings of the Berkeley Actuarial Research Conference on Credibility, Academic Press.

`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,
  data,
  weights,
  p = 1.5,
  link.power = 0,
  muHatGLM = TRUE,
  epsilon = 1e-04,
  maxiter = 500,
  maxiterGLM = 500,
  verbose = FALSE,
  returnData = TRUE,
  balanceProperty = TRUE,
  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, 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 ϵ; the iterations converge when 7 $\|\|\theta[k] - \theta[k-1]\|\|^2[[2]] / \|\|\theta[k-1]\|\|^2[[2]] < \epsilon$. 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.

References

Campo, B.D.C. and Antonio, Katrien (2023). Insurance pricing with hierarchically structured data an illustration with a workers' compensation insurance portfolio. *Scandinavian Actuarial Journal*, doi: 10.1080/03461238.2022.2161413

Ohlsson, E. (2008). Combining generalized linear models and credibility models in practice. *Scandinavian Actuarial Journal* **2008**(4), 301–314.

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

---

hierCredibility *Hierarchical credibility model of Jewell*

Description

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

Usage

```
hierCredibility(
  Yijkt,
  wijkt,
  sector,
  group,
  data,
  muHat = NULL,
  type = c("additive", "multiplicative"),
  returnData = 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. s2 is the estimated variance of the individual contracts, tausq the estimate of $Var(V[j])$ and nusq is the estimate of $Var(V[jk])$.
Means	The estimated averages at the portfolio level (intercept term μ), at the first hierarchical level ($bar(Y)[\%.\%j\%.\%\%.\%]^z$) and at the second hierarchical level ($bar(Y)[\%.\%jk\%.\%]$).
Weights	The weights at the first hierarchical level $z[j\%.\%]$ 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

Campo, B.D.C. and Antonio, Katrien (2023). Insurance pricing with hierarchically structured data an illustration with a workers' compensation insurance portfolio. *Scandinavian Actuarial Journal*, doi: 10.1080/03461238.2022.2161413

Dannenburg, D. R., Kaas, R. and Goovaerts, M. J. (1996). *Practical actuarial credibility models*. Amsterdam: IAE (Institute of Actuarial Science and Econometrics of the University of Amsterdam).

Jewell, W. S. (1975). *The use of collateral data in credibility theory: a hierarchical model*. Laxenburg: IIASA.

Ohlsson, E. (2005). Simplified estimation of structure parameters in hierarchical credibility. *Presented at the Zurich ASTIN Colloquium*.

See Also

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

Examples

```
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

```
## 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 $Var(V[j])$ and nusq is the estimate of $Var(V[jk])$.
Means	The estimated averages at the portfolio level (intercept term μ), at the first hierarchical level ($bar(Y)[\%.\%j\%.\%\%.\%]^z$) and at the second hierarchical level ($bar(Y)[\%.\%jk\%.\%]$).
Weights	The weights at the first hierarchical level $z[j\%.\%]$ 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.

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

```
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, `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 ϵ; the iterations converge when $||\theta[k] - \theta[k-1]||^2[[2]]/||\theta[k-1]||^2[[2]] < \epsilon$. 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.
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

Ohlsson, E. (2008). Combining generalized linear models and credibility models in practice. *Scandinavian Actuarial Journal* **2008**(4), 301–314.

See Also

hierCredTweedie-class, fitted.hierCredTweedie, predict.hierCredTweedie, ranef-actuaRE, weights-actuaRE, hierCredibility, hierCredGLM, cpglm, plotRE, adjustIntercept, BalanceProperty
@references Campo, B.D.C. and Antonio, Katrien (2023). Insurance pricing with hierarchically structured data an illustration with a workers' compensation insurance portfolio. *Scandinavian Actuarial Journal*, doi: 10.1080/03461238.2022.2161413

Examples

```
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

```
## 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 *Formula*

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 = formula(y ~ x)
is.formula(f)
```

`isNested` *Is f1 nested within f2?*

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

```
library(lme4)
with(Pastes, isNested(cask, batch))   ## => FALSE
with(Pastes, isNested(sample, batch))  ## => TRUE
```

modular *Modular Functions for Mixed Model Fits*

Description

Modular functions for mixed model fits

Usage

```
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 (*such methods are not guaranteed to work properly if* data *is omitted*). 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 **for** [g]lFormula**:** 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

```
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

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

NrUnique *Number of unique elements in a vector*

Description

Number of unique elements in a vector

Usage

```
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

```
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

```
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.

Examples

```
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

```
## 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`

print.BalanceProperty *Print method for an object of class* BalanceProperty

Description

Print method for an object of class BalanceProperty

Usage

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

Arguments

x	an object of type BalanceProperty
...	Currently ignored.

Value

Prints the call and whether the balance property is satisfied or not. Returns an invisible copy of the original object.

See Also

BalanceProperty

ranef *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 ranef is a data frame with k rows and j columns.

Value

- From `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

```
library(lattice) ## for dotplot, qqmath
library(lme4)
fm1 <- lmer(Reaction ~ Days + (Days|Subject), sleepstudy)
fm2 <- lmer(Reaction ~ Days + (1|Subject) + (0+Days|Subject), sleepstudy)
fm3 <- lmer(diameter ~ (1|plate) + (1|sample), Penicillin)
ranef(fm1)
```

---

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.

Usage

```
## 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.

simulatedclustereddata *Simulated data sets to illustrate the package functionality*

Description

Both the tweedietraindata and tweedietestdata dataframe are synthetically generated data sets to illustrate the functionality of the package. The tweedietraindata has 250 000 observations and the tweedietestdata has 250 000 observations. The same settings were used to generate both data sets.

Usage

```
data(tweedietraindata)
data(tweedietestdata)
```

Format

y the tweedie distributed outcome variable

cluster the cluster

subcluster the subcluster nested within cluster

x1 covariate 1

x2 covariate 2

x3 covariate 3

x4 covariate 4

x5 covariate 5

Details

See the examples for how the data sets were generated.

Examples

```
# The data sets were generated as follows
lapply(c("magrittr", "dplyr", "data.table", "tweedie"), library, character.only = TRUE)
set.seed(1)

# Simulate training data
set.seed(1)
nClusters    = 5
nSubclusters = 5
p            = 5
Uj           = scale(rnorm(nClusters))
Ujk       = do.call("c", lapply(seq_len(nClusters), function(x) scale(rnorm(nSubclusters))))
nPop         = 1e6
nSample      = 50
```

```
nTest        = 1e3
X            = replicate(p, rnorm(nPop))
Beta         = rnorm(p)
cluster      = sample(seq_len(nClusters), nPop, TRUE)
subcluster   = NULL
uniqueCl     = sort(unique(cluster))
for(cl in uniqueCl)
  subcluster[cluster == cl] <- sample(
    1 - seq_len(nSubclusters) + which(cl == uniqueCl) * nSubclusters,
    sum(cluster == cl),
    TRUE)
table(subcluster, cluster)
eta       = X %*% Beta + Uj[match(cluster, seq_len(nClusters))] +
            Ujk[match(subcluster, seq_len(nClusters * nSubclusters))]
y         = rtweedie(nPop, mu = exp(as.vector(eta)), phi = 1, power = 1.5)
wt        = runif(nPop)
Dt        = data.frame(y, X, wt, cluster, subcluster)
colnames(Dt) %<>% tolower

tweedietraindata = Dt %>%
  group_by(subcluster) %>%
  sample_n(size = nSample) %>%
  as.data.table

tweedietestdata = Dt %>%
  group_by(subcluster) %>%
  sample_n(size = nSample) %>%
  as.data.table
```

`tweedieGLMM` *Fitting a Tweedie GLMM, using the initial estimates of hierCredTweedie*

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

```
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 ϵ; the iterations converge when $\|\|\theta[k] - \theta[k-1]\|\|^2[[2]]/\|\|\theta[k-1]\|\|^2[[2]] < \epsilon$. 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 the balance property should be satisfied.

Value

an object of class cpglmm, containing the model fit.

References

Campo, B.D.C. and Antonio, Katrien (2023). Insurance pricing with hierarchically structured data an illustration with a workers' compensation insurance portfolio. *Scandinavian Actuarial Journal*, doi: 10.1080/03461238.2022.2161413

See Also

cpglmm and hierCredTweedie

Examples

```
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

```
## 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