Package ‘cAIC4’

April 4, 2025

Type Package

Title Conditional Akaike Information Criterion for 'lme4' and 'nlme'

Version 1.1

Date 2025-04-04

Depends lme4(>= 1.1-6), methods, Matrix, stats4, nlme

Imports RLRsim, mgcv, mvtnorm

Suggests gamm4

Description Provides functions for the estimation of the conditional Akaike information in generalized mixed-effect models fitted with (g)lmer() from 'lme4', lme() from 'nlme' and gamm() from 'mgcv'. For a manual on how to use 'cAIC4', see Saefken et al. (2021) <doi:10.18637/jss.v099.i08>.

License GPL (>= 2)

NeedsCompilation no

Date/Publication 2025-04-04 00:10:02 UTC

RoxygenNote 7.3.2

Encoding UTF-8

Author Benjamin Saefken [aut], David Ruegamer [aut, cre], Philipp Baumann [aut], Rene-Marcel Kruse [aut], Sonja Greven [aut], Thomas Kneib [aut]

Maintainer David Ruegamer <david.ruegamer@gmail.com>

Repository CRAN

Contents

`cAIC4-package` *Conditional Akaike Information Criterion for 'lme4' and 'nlme'*

Description

Provides functions for the estimation of the conditional Akaike information in generalized mixed-effect models fitted with (g)lmer() from 'lme4', lme() from 'nlme' and gamm() from 'mgcv'. For a manual on how to use 'cAIC4', see Saefken et al. (2021) <doi:10.18637/jss.v099.i08>.

Details

The DESCRIPTION file:

Package:	cAIC4
Type:	Package
Title:	Conditional Akaike Information Criterion for 'lme4' and 'nlme'
Version:	1.1
Date:	2025-04-04
Authors@R:	c(person(given = "Benjamin", family = "Saefken", role = "aut"), person(given = "David", family = "Rue
Depends:	lme4(>= 1.1-6), methods, Matrix, stats4, nlme
Imports:	RLRsim, mgcv, mvtnorm
Suggests:	gamm4
Description:	Provides functions for the estimation of the conditional Akaike information in generalized mixed-effect
License:	GPL (>= 2)
Packaged:	2021-09-22 12:34:56 UTC; david
NeedsCompilation:	no
Date/Publication:	2014-08-12 11:48:10
RoxygenNote:	7.3.2
Encoding:	UTF-8
Author:	Benjamin Saefken [aut], David Ruegamer [aut, cre], Philipp Baumann [aut], Rene-Marcel Kruse [aut], S
Maintainer:	David Ruegamer <david.ruegamer@gmail.com>

Index of help topics:

```
Zambia                  Subset of the Zambia data set on childhood
                        malnutrition
anocAIC                 Comparison of several lmer objects via cAIC
cAIC                    Conditional Akaike Information for 'lme4' and
                        'lme'
cAIC4-package           Conditional Akaike Information Criterion for
                        'lme4' and 'nlme'
deleteZeroComponents    Delete random effect terms with zero variance
family.lme              family function for lme objects to have a
                        generic function
getModelComponents      Generic getModelComponents method
getModelComponents.lme
                        getModelComponents for lme objects
getModelComponents.merMod
                        getModelComponents for merMods
getWeights              Optimize weights for model averaging.
getcondLL               Function to calculate the conditional
                        log-likelihood
guWahbaData             Data from Gu and Wahba (1991)
modelAvg                Model Averaging for Linear Mixed Models
predictMA               Prediction of model averaged linear mixed
                        models
print.cAIC              Print method for cAIC
stepcAIC                Function to stepwise select the (generalized)
                        linear mixed model fitted via (g)lmer() or
                        (generalized) additive (mixed) model fitted via
                        gamm4() with the smallest cAIC.
summaryMA               Summary of model averaged linear mixed models
```

Author(s)

Benjamin Saefken [aut], David Ruegamer [aut, cre], Philipp Baumann [aut], Rene-Marcel Kruse [aut], Sonja Greven [aut], Thomas Kneib [aut]

Maintainer: David Ruegamer <david.ruegamer@gmail.com>

References

Saefken, B., Kneib T., van Waveren C.-S. and Greven, S. (2014) A unifying approach to the estimation of the conditional Akaike information in generalized linear mixed models. Electronic Journal Statistics Vol. 8, 201-225.

Greven, S. and Kneib T. (2010) On the behaviour of marginal and conditional AIC in linear mixed models. Biometrika 97(4), 773-789.

Efron , B. (2004) The estimation of prediction error. J. Amer. Statist. Ass. 99(467), 619-632.

See Also

lme4

Examples

```
b <- lmer(Reaction ~ Days + (Days | Subject), sleepstudy)

cAIC(b)
```

---

anocAIC — *Comparison of several lmer objects via cAIC*

Description

Takes one or more `lmer`-objects and produces a table to the console.

Usage

```
anocAIC(object, ..., digits = 2)
```

Arguments

`object`	a fitted `lme4`-object
`...`	additional objects of the same type
`digits`	number of digits to print

Value

a table comparing the cAIC relevant information of all models

See Also

`cAIC` for the model fit.

---

cAIC — *Conditional Akaike Information for 'lme4' and 'lme'*

Description

Estimates the conditional Akaike information for models that were fitted in 'lme4' or with 'lme'. Currently all distributions are supported for 'lme4' models, based on parametric conditional bootstrap. For the Gaussian distribution (from a `lmer` or `lme` call) and the Poisson distribution analytical estimators for the degrees of freedom are available, based on Stein type formulas. Also the conditional Akaike information for generalized additive models based on a fit via the 'gamm4' or `gamm` calls from the 'mgcv' package can be estimated. A hands-on tutorial for the package can be found at `https://arxiv.org/abs/1803.05664`.

Usage

```
cAIC(object, method = NULL, B = NULL, sigma.penalty = 1, analytic = TRUE)
```

Arguments

- `object` — An object of class merMod either fitted by `lmer` or `glmer` of the lme4-package or an `lme` object fro the nlme-package. Also objects returned form a `gamm4` call are possible.
- `method` — Either `"conditionalBootstrap"` for the estimation of the degrees of freedom with the help of conditional Bootstrap or `"steinian"` for analytical representations based on Stein type formulas. The default is `NULL`. In this case the method is choosen automatically based on the `family` argument of the `(g)lmer`-object. For `"gaussian"`, `"poisson"` and `"binomial"` (in the Bernoulli case, i.e. only two outcomes) this is the Steinian type estimator, for all others it is the conditional Bootstrap. For models from the nlme package, only `lme` objects, i.e., with gaussian response are supported.
- `B` — Number of Bootstrap replications. The default is `NULL`. Then B is the minimum of 100 and the length of the response vector.
- `sigma.penalty` — An integer value for additional penalization in the analytic Gaussian calculation to account for estimated variance components in the residual (co-)variance. Per default `sigma.penalty` is equal `1`, corresponding to a diagonal error covariance matrix with only one estimated parameter (sigma). If all variance components are known, the value should be set to `0`. For individual weights (individual variances), this value should be set to the number of estimated weights. For `lme` objects the penalty term is automatically set by extracting the number of estimated variance components.
- `analytic` — FALSE if the numeric hessian of the (restricted) marginal log-likelihood from the lmer optimization procedure should be used. Otherwise (default) TRUE, i.e. use a analytical version that has to be computed. Only used for the analytical version of Gaussian responses.

Details

For `method = "steinian"` and an object of class `merMod` computed the analytic representation of the corrected conditional AIC in Greven and Kneib (2010). This is based on a the Stein formula and uses implicit differentiation to calculate the derivative of the random effects covariance parameters w.r.t. the data. The code is adapted form the one provided in the supplementary material of the paper by Greven and Kneib (2010). The supplied `merMod` model needs to be checked if a random effects covariance parameter has an optimum on the boundary, i.e. is zero. And if so the model needs to be refitted with the according random effect terms omitted. This is also done by the function and the refitted model is also returned. Notice that the `boundary.tol` argument in `lmerControl` has an impact on whether a parameter is estimated to lie on the boundary of the parameter space. For estimated error variance the degrees of freedom are increased by one per default. `sigma.penalty` can be set manually for `merMod` models if no (0) or more than one variance component (>1) has been estimated. For `lme` objects this value is automatically defined.

If the object is of class `merMod` and has `family = "poisson"` there is also an analytic representation of the conditional AIC based on the Chen-Stein formula, see for instance Saefken et. al (2014). For the calculation the model needs to be refitted for each observed response variable minus the number of response variables that are exactly zero. The calculation therefore takes longer then for models with Gaussian responses. Due to the speed and stability of 'lme4' this is still possible, also for larger datasets.

If the model has Bernoulli distributed responses and method = "steinian", cAIC calculates the degrees of freedom based on a proposed estimator by Efron (2004). This estimator is asymptotically unbiased if the estimated conditional mean is consistent. The calculation needs as many model refits as there are data points.

Another more general method for the estimation of the degrees of freedom is the conditional bootstrap. This is proposed in Efron (2004). For the B boostrap samples the degrees of freedom are estimated by

$$\frac{1}{B-1}\sum_{i=1}^{n}\theta_i(z_i)(z_i-\bar{z}),$$

where $\theta_i(z_i)$ is the i-th element of the estimated natural parameter.

For models with no random effects, i.e. (g)lms, the cAIC function returns the AIC of the model with scale parameter estimated by REML.

Value

A cAIC object, which is a list consisting of: 1. the conditional log likelihood, i.e. the log likelihood with the random effects as penalized parameters; 2. the estimated degrees of freedom; 3. a list element that is either NULL if no new model was fitted otherwise the new (reduced) model, see details; 4. a boolean variable indicating whether a new model was fitted or not; 5. the estimator of the conditional Akaike information, i.e. minus twice the log likelihood plus twice the degrees of freedom.

WARNINGS

Currently the cAIC can only be estimated for family equal to "gaussian", "poisson" and "binomial". Neither negative binomial nor gamma distributed responses are available. Weighted Gaussian models are not yet implemented.

Author(s)

Benjamin Saefken, David Ruegamer

References

Saefken, B., Ruegamer, D., Kneib, T. and Greven, S. (2021): Conditional Model Selection in Mixed-Effects Models with cAIC4. <doi:10.18637/jss.v099.i08>

Saefken, B., Kneib T., van Waveren C.-S. and Greven, S. (2014) A unifying approach to the estimation of the conditional Akaike information in generalized linear mixed models. Electronic Journal Statistics Vol. 8, 201-225.

Greven, S. and Kneib T. (2010) On the behaviour of marginal and conditional AIC in linear mixed models. Biometrika 97(4), 773-789.

Efron , B. (2004) The estimation of prediction error. J. Amer. Statist. Ass. 99(467), 619-632.

See Also

lme4-package, lmer, glmer

Examples

```
### Three application examples
b <- lmer(Reaction ~ Days + (Days | Subject), sleepstudy)
cAIC(b)

b2 <- lmer(Reaction ~ (1 | Days) + (1 | Subject), sleepstudy)
cAIC(b2)

b2ML <- lmer(Reaction ~ (1 + Days | Subject), sleepstudy, REML = FALSE)
cAIC(b2ML)

### Demonstration of boundary case
## Not run:
set.seed(2017-1-1)
n <- 50
beta <- 2
x <- rnorm(n)
eta <- x*beta
id <- gl(5,10)
epsvar <- 1
data <- data.frame(x = x, id = id)
y_wo_bi <- eta + rnorm(n, 0, sd = epsvar)

# use a very small RE variance
ranvar <- 0.05
nrExperiments <- 100

sim <- sapply(1:nrExperiments, function(j){

b_i <- scale(rnorm(5, 0, ranvar), scale = FALSE)
y <- y_wo_bi + model.matrix(~ -1 + id) %*% b_i
data$y <- y

mixedmod <- lmer(y ~ x + (1 | id), data = data)
linmod <- lm(y ~ x, data = data)

c(cAIC(mixedmod)$caic, cAIC(linmod)$caic)
})

rownames(sim) <- c("mixed model", "linear model")

boxplot(t(sim))



## End(Not run)
```

deleteZeroComponents *Delete random effect terms with zero variance*

Description

Is used in the cAIC function if method = "steinian" and family = "gaussian". The function deletes all random effects terms from the call if corresponding variance parameter is estimated to zero and updates the model in merMod.

Usage

```
deleteZeroComponents(m)

## S3 method for class 'lme'
deleteZeroComponents(m)

## S3 method for class 'merMod'
deleteZeroComponents(m)
```

Arguments

m	An object of class merMod fitted by lmer of the lme4-package or of class lme.

Details

For merMod class models: Uses the cnms slot of m and the relative covariance factors to rewrite the random effects part of the formula, reduced by those parameters that have an optimum on the boundary. This is necessary to obtain the true conditional corrected Akaike information. For the theoretical justification see Greven and Kneib (2010). The reduced model formula is then updated. The function deleteZeroComponents is then called iteratively to check if in the updated model there are relative covariance factors parameters on the boundary.

For lme class models: ...

Value

An updated object of class merMod or of class lme.

NULL

NULL

WARNINGS

For models called via gamm4 or gamm no automated update is available. Instead a warning with terms to omit from the model is returned.

Author(s)

Benjamin Saefken, David Ruegamer, Philipp Baumann

References

Greven, S. and Kneib T. (2010) On the behaviour of marginal and conditional AIC in linear mixed models. Biometrika 97(4), 773-789.

See Also

lme4-package, lmer, getME

Examples

```
## Currently no data with variance equal to zero...
b <- lmer(Reaction ~ Days + (Days | Subject), sleepstudy)

deleteZeroComponents(b)
```

---

family.lme *family function for lme objects to have a generic function*

Description

family function for lme objects to have a generic function

Usage

```
## S3 method for class 'lme'
family(object, ...)
```

Arguments

object	lme object
...	unused

Value

returns a Gaussian distribution

getcondLL	*Function to calculate the conditional log-likelihood*

Description

Function to calculate the conditional log-likelihood

Usage

```
getcondLL(object)

## S3 method for class 'lme'
getcondLL(object)

## S3 method for class 'merMod'
getcondLL(object)
```

Arguments

`object`	An object of class `merMod` either fitted by `lmer` or `glmer` of the 'lme4' package.

Value

conditional log-likelihood value

NULL

NULL

getModelComponents	*Generic getModelComponents method*

Description

Generic getModelComponents method

Usage

```
getModelComponents(m, analytic)
```

Arguments

`m`	model object
`analytic`	logical

Value

Model components

`getModelComponents.lme`

getModelComponents for lme objects

Description

getModelComponents for lme objects

Usage

```
## S3 method for class 'lme'
getModelComponents(m, analytic = TRUE)
```

Arguments

`m`	lme object
`analytic`	logical

Value

Model components

`getModelComponents.merMod`

getModelComponents for merMods

Description

getModelComponents for merMods

Usage

```
## S3 method for class 'merMod'
getModelComponents(m, analytic)
```

Arguments

`m`	merMod object
`analytic`	logical

`getWeights` *Optimize weights for model averaging.*

Description

Function to determine optimal weights for model averaging based on a proposal by Zhang et al. (2014) to derive a weight choice criterion based on the conditional Akaike Information Criterion as proposed by Greven and Kneib (2010). The underlying optimization is a customized version of the Augmented Lagrangian Method.

Usage

```
getWeights(models)
```

Arguments

`models`	An list object containing all considered candidate models fitted by `lmer` of the lme4-package or of class `lme`.

Value

An object containing a vector of optimized weights, value of the minimized target function and the duration of the optimization process.

WARNINGS

No weight-determination is currently possible for models called via `gamm4`.

Author(s)

Benjamin Saefken & Rene-Marcel Kruse

References

Greven, S. and Kneib T. (2010) On the behaviour of marginal and conditional AIC in linear mixed models. Biometrika 97(4), 773-789.

Zhang, X., Zou, G., & Liang, H. (2014). Model averaging and weight choice in linear mixed-effects models. Biometrika, 101(1), 205-218.

Nocedal, J., & Wright, S. (2006). Numerical optimization. Springer Science & Business Media.

See Also

`lme4-package`, `lmer`, `getME`

Examples

```
data(Orthodont, package = "nlme")
models <- list(
    model1 <- lmer(formula = distance ~ age + Sex + (1 | Subject) + age:Sex,
               data = Orthodont),
    model2 <- lmer(formula = distance ~ age + Sex + (1 | Subject),
               data = Orthodont),
    model3 <- lmer(formula = distance ~ age + (1 | Subject),
                 data = Orthodont),
    model4 <- lmer(formula = distance ~ Sex + (1 | Subject),
                data = Orthodont))

foo <- getWeights(models = models)
foo
```

`guWahbaData` *Data from Gu and Wahba (1991)*

Description

Data from Gu and Wahba (1991) which is used for demonstrative purposes to exemplarily fit a generalized additive mixed model.

References

Gu and Wahba (1991) Minimizing GCV/GML scores with multiple smoothing parameters via the Newton method. SIAM J. Sci. Statist. Comput. 12:383-398

`modelAvg` *Model Averaging for Linear Mixed Models*

Description

Function to perform model averaging for linear mixed models based on the weight selection criterion as proposed by Zhang et al. (2014).

Usage

```
modelAvg(models, opt = TRUE)
```

Arguments

models — A list object containing all considered candidate models fitted by `lmer` of the lme4-package or of class `lme`.

opt — logical. If TRUE (the default) the model averaging approach based on Zhang et al. is applied. If FALSE the underlying weights are calculated as smoothed weights as proposed by Buckland et al. (1997).

Value

An object containing the function calls of the underlying candidate models, the values of the model averaged fixed effects, the values of the model averaged random effects, the results of the weight optimization process, as well as a list of the candidate models themselves.

Author(s)

Benjamin Saefken & Rene-Marcel Kruse

References

Greven, S. and Kneib T. (2010) On the behaviour of marginal and conditional AIC in linear mixed models. Biometrika 97(4), 773-789.

Zhang, X., Zou, G., & Liang, H. (2014). Model averaging and weight choice in linear mixed-effects models. Biometrika, 101(1), 205-218.

See Also

`lme4-package`, `lmer`

Examples

```
data(Orthodont, package = "nlme")
models <- list(
    model1 <- lmer(formula = distance ~ age + Sex + (1 | Subject) + age:Sex,
                   data = Orthodont),
    model2 <- lmer(formula = distance ~ age + Sex + (1 | Subject),
                   data = Orthodont),
    model3 <- lmer(formula = distance ~ age + (1 | Subject),
                     data = Orthodont),
    model4 <- lmer(formula = distance ~ Sex + (1 | Subject),
                    data = Orthodont))
foo <- modelAvg(models = models)
foo
```

predictMA *Prediction of model averaged linear mixed models*

Description

Function to perform prediction for model averaged linear mixed models based on the weight selection criterion as proposed by Zhang et al.(2014)

Usage

```
predictMA(object, new.data)
```

Arguments

object	A object created by the model averaging function.
new.data	Object that contains the data on which the prediction is to be based on.

Value

An object that contains predictions calculated based on the given dataset and the assumed underlying model average.

Author(s)

Benjamin Saefken & Rene-Marcel Kruse

References

Greven, S. and Kneib T. (2010) On the behaviour of marginal and conditional AIC in linear mixed models. Biometrika 97(4), 773-789.

See Also

lme4-package, lmer

Examples

```
data(Orthodont, package = "nlme")
models <- list(
    model1 <- lmer(formula = distance ~ age + Sex + (1 | Subject) + age:Sex,
                data = Orthodont),
    model2 <- lmer(formula = distance ~ age + Sex + (1 | Subject),
                data = Orthodont),
    model3 <- lmer(formula = distance ~ age + (1 | Subject),
                  data = Orthodont),
    model4 <- lmer(formula = distance ~ Sex + (1 | Subject),
                 data = Orthodont))
foo <- modelAvg(models = models)
predictMA(foo, new.data = Orthodont)
```

`print.cAIC` *Print method for cAIC*

Description

Print method for cAIC

Usage

```
## S3 method for class 'cAIC'
print(x, ..., digits = 2)
```

Arguments

`x`	a cAIC object
`...`	further arguments passed to generic print function (not in use).
`digits`	number of digits to print

`stepcAIC` *Function to stepwise select the (generalized) linear mixed model fitted via (g)lmer() or (generalized) additive (mixed) model fitted via gamm4() with the smallest cAIC.*

Description

The step function searches the space of possible models in a greedy manner, where the direction of the search is specified by the argument direction. If direction = "forward" / = "backward", the function adds / exludes random effects until the cAIC can't be improved further. In the case of forward-selection, either a new grouping structure, new slopes for the random effects or new covariates modeled nonparameterically must be supplied to the function call. If direction = "both", the greedy search is alternating between forward and backward steps, where the direction is changed after each step

Usage

```
stepcAIC(
  object,
  numberOfSavedModels = 1,
  groupCandidates = NULL,
  slopeCandidates = NULL,
  fixEfCandidates = NULL,
  numberOfPermissibleSlopes = 2,
```

```
  allowUseAcross = FALSE,
  allowCorrelationSel = FALSE,
  allowNoIntercept = FALSE,
  direction = "backward",
  trace = FALSE,
  steps = 50,
  keep = NULL,
  numCores = 1,
  data = NULL,
  returnResult = TRUE,
  calcNonOptimMod = TRUE,
  bsType = "tp",
  digits = 2,
  printValues = "caic",
  ...
)
```

Arguments

`object` object returned by `[lme4]{lmer}`, `[lme4]{glmer}` or `[gamm4]{gamm4}`

`numberOfSavedModels` integer defining how many additional models to be saved during the step procedure. If `1` (DEFAULT), only the best model is returned. Any number `k` greater `1` will return the `k` best models. If `0`, all models will be returned (not recommended for larger applications).

`groupCandidates` character vector containing names of possible grouping variables for new random effects. Group nesting must be specified manually, i.e. by listing up the string of the groups in the manner of lme4. For example `groupCandidates = c("a", "b", "a/b")`.

`slopeCandidates` character vector containing names of possible new random effects

`fixEfCandidates` character vector containing names of possible (non-)linear fixed effects in the GAMM; NULL for the (g)lmer-use case

`numberOfPermissibleSlopes` how much slopes are permissible for one grouping variable

`allowUseAcross` allow slopes to be used in other grouping variables

`allowCorrelationSel` logical; FALSE does not allow correlations of random effects to be (de-)selected (default)

`allowNoIntercept` logical; FALSE does not allow random effects without random intercept

`direction` character vector indicating the direction ("both","backward","forward")

`trace` logical; should information be printed during the execution of stepcAIC?

`steps` maximum number of steps to be considered

keep	list($fixed,$random) of formulae; which splines / fixed (fixed) or random effects (random) to be kept during selection; specified terms must be included in the original model
numCores	the number of cores to be used in calculations; parallelization is done by using `parallel::mclapply`
data	data.frame supplying the data used in `object`. `data` must also include variables, which are considered for forward updates.
returnResult	logical; whether to return the result (best model and corresponding cAIC)
calcNonOptimMod	logical; if FALSE, models which failed to converge are not considered for cAIC calculation
bsType	type of splines to be used in forward gamm4 steps
digits	number of digits used in printing the results
printValues	what values of `c("cll", "df", "caic", "refit")` to print in the table of comparisons
...	further options for cAIC call

Value

if `returnResult` is `TRUE`, a list with the best model `finalModel`, `additionalModels` if `numberOfSavedModels` was specified and the corresponding cAIC `bestCAIC` is returned.

Note that if `trace` is set to `FALSE` and `returnResult` is also `FALSE`, the function call may not be meaningful

Details

Note that the method can not handle mixed models with uncorrelated random effects and does NOT reduce models to such, i.e., the model with `(1 + s | g)` is either reduced to `(1 | g)` or `(0 + s | g)` but not to `(1 + s || g)`.

Author(s)

David Ruegamer

Examples

```
(fm3 <- lmer(strength ~ 1 + (1|sample) + (1|batch), Pastes))

fm3_step <- stepcAIC(fm3, direction = "backward", trace = TRUE, data = Pastes)

fm3_min <- lm(strength ~ 1, data=Pastes)

fm3_min_step <- stepcAIC(fm3_min, groupCandidates = c("batch", "sample"),
direction="forward", data=Pastes, trace=TRUE)
fm3_min_step <- stepcAIC(fm3_min, groupCandidates = c("batch", "sample"),
direction="both", data=Pastes, trace=TRUE)
# try using a nested group effect which is actually not nested -> warning
fm3_min_step <- stepcAIC(fm3_min, groupCandidates = c("batch", "sample", "batch/sample"),
```

```
                         direction="both", data=Pastes, trace=TRUE)

Pastes$time <- 1:dim(Pastes)[1]
fm3_slope <- lmer(data=Pastes, strength ~ 1 + (1 + time | cask))

fm3_slope_step <- stepcAIC(fm3_slope,direction="backward", trace=TRUE, data=Pastes)



fm3_min <- lm(strength ~ 1, data=Pastes)

fm3_min_step <- stepcAIC(fm3_min,groupCandidates=c("batch","sample"),
direction="forward", data=Pastes,trace=TRUE)



fm3_inta <- lmer(strength ~ 1 + (1|sample:batch), data=Pastes)

fm3_inta_step <- stepcAIC(fm3_inta,groupCandidates=c("batch","sample"),
direction="forward", data=Pastes,trace=TRUE)

fm3_min_step2 <- stepcAIC(fm3_min,groupCandidates=c("cask","batch","sample"),
direction="forward", data=Pastes,trace=TRUE)

fm3_min_step3 <- stepcAIC(fm3_min,groupCandidates=c("cask","batch","sample"),
direction="both", data=Pastes,trace=TRUE)

## Not run:
fm3_inta_step2 <- stepcAIC(fm3_inta,direction="backward",
data=Pastes,trace=TRUE)

## End(Not run)

##### create own example


na <- 20
nb <- 25
n <- 400
a <- sample(1:na,400,replace=TRUE)
b <- factor(sample(1:nb,400,replace=TRUE))
x <- runif(n)
y <- 2 + 3 * x + a*.02 + rnorm(n) * .4
a <- factor(a)
c <- interaction(a,b)
y <- y + as.numeric(as.character(c))*5
df <- data.frame(y=y,x=x,a=a,b=b,c=c)

smallMod <- lm(y ~ x)

## Not run:
# throw error
stepcAIC(smallMod, groupCandidates=c("a","b","c"), data=df, trace=TRUE, returnResult=FALSE)
```

```
smallMod <- lm(y ~ x, data=df)

# throw error
stepcAIC(smallMod, groupCandidates=c("a","b","c"), data=df, trace=TRUE, returnResult=FALSE)

# get it all right
mod <- stepcAIC(smallMod, groupCandidates=c("a","b","c"),
                data=df, trace=TRUE,
                direction="forward", returnResult=TRUE)

# make some more steps...
stepcAIC(smallMod, groupCandidates=c("a","b","c"), data=df, trace=TRUE,
         direction="both", returnResult=FALSE)

mod1 <- lmer(y ~ x + (1|a), data=df)

stepcAIC(mod1, groupCandidates=c("b","c"), data=df, trace=TRUE, direction="forward")
stepcAIC(mod1, groupCandidates=c("b","c"), data=df, trace=TRUE, direction="both")



mod2 <- lmer(y ~ x + (1|a) + (1|c), data=df)

stepcAIC(mod2, data=df, trace=TRUE, direction="backward")

mod3 <- lmer(y ~ x + (1|a) + (1|a:b), data=df)

stepcAIC(mod3, data=df, trace=TRUE, direction="backward")


## End(Not run)
```

`summaryMA` *Summary of model averaged linear mixed models*

Description

summaryMA is a function used to produce result summaries of the model averaging approach.

Usage

```
summaryMA(object, randeff = FALSE)
```

Arguments

`object`	A object created by the model averaging function.
`randeff`	logical. Indicator whether the model averaged random effects should also be part of the output. The default setting is FALSE.

Value

Outputs a summary of the model averaged random and fixed effects, as well as the calculated weights of the individual candidate models.

Author(s)

Benjamin Saefken & Rene-Marcel Kruse

References

Greven, S. and Kneib T. (2010) On the behaviour of marginal and conditional AIC in linear mixed models. Biometrika 97(4), 773-789.

See Also

lme4-package, lmer

Examples

```
data(Orthodont, package = "nlme")
models <- list(
    model1 <- lmer(formula = distance ~ age + Sex + (1 | Subject) + age:Sex,
               data = Orthodont),
    model2 <- lmer(formula = distance ~ age + Sex + (1 | Subject),
               data = Orthodont),
    model3 <- lmer(formula = distance ~ age + (1 | Subject),
                 data = Orthodont),
    model4 <- lmer(formula = distance ~ Sex + (1 | Subject),
                data = Orthodont))
foo <- modelAvg(models = models)
summaryMA(foo)
```

Zambia — *Subset of the Zambia data set on childhood malnutrition*

Description

Data analyzed by Kandala et al. (2001) which is used for demonstrative purposes to estimate linear mixed and additive models using a stepwise procedure on the basis of the cAIC. The full data set is available at https://www.uni-goettingen.de/de/551625.html.

References

Kandala, N. B., Lang, S., Klasen, S., Fahrmeir, L. (2001): Semiparametric Analysis of the Socio-Demographic and Spatial Determinants of Undernutrition in Two African Countries. Research in Official Statistics, 1, 81-100.

Index