Package 'asremlPlus'

September 16, 2016

Version 2.0-12

Date 2016-09-16

Title Augments the Use of 'ASReml-R' in Fitting Mixed Models

Author Chris Brien <Chris.Brien@unisa.edu.au>.

Maintainer Chris Brien <Chris.Brien@unisa.edu.au>

Depends R (>= 2.10.0)

Imports dae, ggplot2, stats, methods, utils, grDevices

Suggests testthat

Enhances asreml

SystemRequirements asreml-R 3.x

Description Assists in automating the testing of terms in mixed models when 'asreml' is used to fit the models. The content falls into the following natural groupings: (i) Data, (ii) Object manipulation functions, (iii) Model modification functions, (iv) Model testing functions, (v) Model diagnostics functions, (vi) Prediction production and presentation functions, (vii) Response transformation functions, and (viii) Miscellaneous functions. A history of the fitting of a sequence of models is kept in a data frame. Procedures are available for choosing models that conform to the hierarchy or marginality principle and for displaying predictions for significant terms in tables and graphs. The package 'asreml' provides a computationally efficient algorithm for fitting mixed models using Residual Maximum Likelihood. It can be purchased from 'VSNi' <http://www.vsni.co.uk/> as 'asreml-R', who will supply a zip file for local installation/updating.

License GPL (>= 2)

URL http://chris.brien.name

NeedsCompilation no

Repository CRAN

Date/Publication 2016-09-16 12:50:20

R topics documented:

`asremlPlus-package` *Augments the Use of 'ASReml-R' in Fitting Mixed Models*

Description

Assists in automating the testing of terms in mixed models when 'asreml' is used to fit the models. The content falls into the following natural groupings: (i) Data, (ii) Object manipulation functions, (iii) Model modification functions, (iv) Model testing functions, (v) Model diagnostics functions, (vi) Prediction production and presentation functions, (vii) Response transformation functions, and (viii) Miscellaneous functions. A history of the fitting of a sequence of models is kept in a data frame. Procedures are available for choosing models that conform to the hierarchy or marginality principle and for displaying predictions for significant terms in tables and graphs. The package

'asreml' provides a computationally efficient algorithm for fitting mixed models using Residual Maximum Likelihood. It can be purchased from 'VSNi' <http://www.vsni.co.uk/> as 'asreml-R', who will supply a zip file for local installation/updating.

Version: 2.0-12

Date: 2016-09-16

Index

(i) Data

Wheat.dat	Data for an experiment to investigate 25 varieties of wheat.
WaterRunoff.dat	Data for an experiment to investigate the quality of water runoff over time

(ii) Object manipulation

alldiffs	Forms an object of S3-class 'alldiffs' that stores the predictions for a model fitted using asreml, along with statistics for all pairwise differences.
asrtests	Forms an object of S3-class 'asrtests' that stores a fitted asreml object, a pseudo-anova table for the fixed and a history of changes and hypthesis testing used in obtaining the model.
print.alldiffs	Prints the values in an 'alldiffs' object in a nice format.
print.asrtests	Prints the values in an 'asrtests' object.

(iii) Model modification

addrm.terms.asrtests	Adds or removes the specified set terms from either the fixed or random model and records the change in a data.frame.
newfit.asreml	Refits an asreml model with modified model formula using either a call to 'update.asreml' or a direct call to 'asreml'.
newrcov.asrtests	Fits a new rcov formula using asreml.
rmboundary.asrtests	Removes any boundary or singular variance components from the fit stored in 'asreml.obj' and records their removal in a data.frame.
setvarianceterms.asreml	Allows the seting of constraints and initial values for terms in the 'random' and 'rcov' arguments of an 'asreml' call.
sig.devn.reparam.asrtests	This function reparamterizes each random (deviations) term involving 'devn.fac' to a fixed term and ensures that the same term, with 'trend.num' replacing 'devn.fac', is included if any other term with 'trend.num' is included in 'terms'.

(iv) Model testing

choose.model.asrtests	Determines the set of significant terms taking into account hierarchy or marginality relations.
info.crit.asreml	Computes AIC and BIC for a model.
recalc.wald.tab.asrtests	Recalculates the denDF, F.inc and P values for a table of Wald test statistics obtained using 'wald.asreml'.
reml.lrt.asreml	Performs REML likelihood ratio test.
testranfix.asrtests	Tests for a single fixed or random term in model fitted using 'asreml'.
testrcov.asrtests	Fits a new rcov formula using 'asreml' and tests whether the change is significant.
testswapran.asrtests	Tests, using a REMLRT, whether the difference between current random model and one in which oldterms are dropped and newterms are added is significant.

(v) Model diagnostics

variofaces.asreml	Plot empirical variogram faces, including envelopes, as described by Stefanova, Smith & Cullis (2009).

(vi) Prediction production and presentation

pred.present.asreml	This function forms the predictions for each significant term and presents them in tables and/or graphs.
predictiondiffs.asreml	Forms all pairwise differences between a set of predictions, their standard errors and p-values for a test of whether the differences are significantly different from zero.
predictionplot.asreml	This function plots the predictions for a term, possibly with error bars.
predictparallel.asreml	This function forms the predictions and associated statistics for a term, taking into account that a numeric vector and a factor having parallel values may occur in the model. It stores the results in an object of class 'alldifffs' and prints the results. It can be used when there are not parallel values.

(vii) Response transformation

angular	Applies the angular transformation to proportions.
angular.mod	Applies the modified angular transformation to a vector of counts.
power.transform	Perform linear and power transformations on a variable whose name is given as a character string in 'var.name'. The transformed variable is stored in the 'data.frame data'.

(viii) Miscellaneous

num.recode	Recodes the unique values of a vector using the values in a new vector.
permute.square	Permutes the rows and columns of a square matrix.
permute.to.zero.lowertri	Permutes a square matrix until all the lower triangular elements are zero.

The functions whose names end in 'asrtests', which are most of the model functions, utilize an asrtests object that stores: (i) the currently fitted model in asreml.obj, (ii) the table of test statistics for the fixed effects in wald.tab, and (iii) a data frame that contains a history of the changes made to the model in test.summary.

Author(s)

Chris Brien <Chris.Brien@unisa.edu.au>.

Maintainer: Chris Brien <Chris.Brien@unisa.edu.au>

References

Butler, D. G., et al. (2010). *Analysis of Mixed Models for S language environments: ASReml-R reference manual.* Brisbane, DPI Publications.

Stefanova, K. T., Smith, A. B. & Cullis, B. R. (2009) Enhanced diagnostics for the spatial analysis of field trials. *Journal of Agricultural, Biological, and Environmental Statistics*, **14**, 392–410.

See Also

asreml

Examples

```
## Not run:
## Analyse wheat dat using asreml and asremlPlus
## Set up for analysis
library(dae)
library(asreml)
## use ?Wheat.dat for data set details
data(Wheat.dat)

# Fit initial model
current.asr <- asreml(yield ~ Rep + WithinColPairs + Variety,
                      random = ~ Row + Column + units,
                      rcov = ~ ar1(Row):ar1(Column),
                      data=Wheat.dat)
summary(current.asr)

# Load current fit into an asrtests object
current.asrt <- asrtests(current.asr, NULL, NULL)
```

```
# Check for and remove any boundary terms
current.asrt <- rmboundary.asrtests(current.asrt)

#Check term for within Column pairs
current.asrt <- testranfix.asrtests("WithinColPairs", current.asrt, drop.fix.ns=TRUE)

# Test nugget term
current.asrt <- testranfix.asrtests("units", current.asrt, positive=TRUE)

# Test Row autocorrelation
current.asrt <- testrcov.asrtests("~ Row:ar1(Column)", current.asrt,
                                  label="Row autocorrelation", simpler=TRUE)

# Test Col autocorrelation (depends on whether Row autocorrelation retained)
k <- match("Row autocorrelation", current.asrt$test.summary$terms)
p <- current.asrt$test.summary$p
{if (p[k] <= 0.05)
  current.asrt <- testrcov.asrtests("~ ar1(Row):Column", current.asrt,
                                    label="Col autocorrelation", simpler=TRUE,
                                    update=FALSE)
 else
   current.asrt <- testrcov.asrtests("~ Row:Column", current.asrt,
                                     label="Col autocorrelation", simpler=TRUE,
                                     update=FALSE)
}
print(current.asrt)

# Get current fitted asreml object
current.asr <- current.asrt$asreml.obj
current.asr <- update(current.asr, aom=TRUE)


# Do residuals-versus-fitted values plot
plot(fitted.values(current.asr),residuals(current.asr))

# Form variance matrix based on estimated variance parameters
s2 <- current.asr$sigma2
gamma.Row <- current.asr$gammas[1]
gamma.unit <- current.asr$gammas[2]
rho.r <- current.asr$gammas[4]
rho.c <- current.asr$gammas[5]
row.ar1 <- mat.ar1(order=10, rho=rho.r)
col.ar1 <- mat.ar1(order=15, rho=rho.c)
V <- fac.vcmat(Wheat.dat$Row, gamma.Row) +
     gamma.unit * diag(1, nrow=150, ncol=150) +
     mat.dirprod(row.ar1, col.ar1)
V <- s2*V

#Produce variogram and variogram faces plot (Stefanaova et al, 2009)
plot.asrVariogram(variogram(current.asr))
variofaces.asreml(current.asr, V=V)
```

```
#Get Variety predictionsand all pairwise prediction differences and p-values
Var.diffs <- predictparallel.asreml(classify = "Variety",
                                    asreml.obj=current.asr,
                                    error.intervals="halfLeast",
                                    wald.tab=current.asrt$wald.tab,
                                    tables = "predictions")
print(Var.diffs, which = c("differences", "p.differences"))

## End(Not run)
```

`addrm.terms.asrtests` *Adds or removes the specified set terms from either the fixed or random model and records the change in a data.frame.*

Description

The specified terms are simply added or removed from either the fixed or random model. No hypothesis testing is performed and no check is made for boundary or singular terms. A row is added to the `test.summary` `data.frame` stating whether fixed or random terms have been added or removed. Convergence in fitting the model is checked and a note included in the action if there was not. All components of the `asrtests` object are updated, although `wald.tab` is only updated if it is present in the supplied `asrtests` object.

Usage

```
addrm.terms.asrtests(terms = NULL, asrtests.obj, add = FALSE,
                     random = FALSE, label = NULL,
                     denDF = "default", trace = FALSE,
                     update = TRUE, set.terms = NULL, ignore.suffices = TRUE,
                     constraints = "P", initial.values = NA, ...)
```

Arguments

`terms`	a single character string in the form of a formula which, after expansion, specifies the sum of a set of terms to be added or dropped.
`asrtests.obj`	an `asrtests` object for a fitted model that is a list containing an `asreml` object, a `wald.tab` `data.frame` with 4 columns, and a `data.frame` with 5 columns that records any previous changes and tests in the fitted model.
`add`	whether to add or remove terms from the model.
`random`	whether terms are to added or removed from the fixed or random model.
`label`	a character string to use as the label in `test.summary` and which indicates what is being tested. If `label` is `NULL`, either `Fixed terms` or `Random terms` is used, depending on whether `random` is `TRUE` or `FALSE`.
`denDF`	Specifies the enthod to use in computing approximate denominator degrees of freedom when `wald.asreml` is called. Can be `none` to suppress the computations, `numeric` for numerical methods, `algebraic` for algebraic methods or

default, the default, to autommatically choose numeric or algebraic computations depending on problem size. The denominator degrees of freedom are calculated according to Kenward and Roger (1997) for fixed terms in the dense part of the model.

trace — if TRUE then partial iteration details are displayed when ASReml-R functions are invoked; if FALSE then no output is displayed.

update — if TRUE then update.asreml is called in removing and adding terms to the model. In doing this the arguments R.param and G.param are set to those in the asreml object stored in the supplied asrtests.obj so that the values from the previous model are used as starting values. If FALSE then calls are made to asreml in which the only changes from the previous call are (i) that the random model is updated and (ii) modifications specified via ... are made.

set.terms — a character vector specifying the terms that are to have constraints and/or initial values set prior to fitting.

ignore.suffices — a logical vector specifying whether the suffices of the asreml-assigned names of the variance terms (i.e. the information to the right of an "!", other than "R!") is to be ignored in matching elements of terms. If TRUE for an element of terms, the suffices are stripped from the asreml-assigned names. If FALSE for an element of terms, the element must exactly match an asreml-assigned name for a variance term. This vector must be of length one or the same length as terms. If it is of length one then the same action is applied to the asreml-assigned suffices for all the terms in terms.

constraints — a character vector specifying the constraints to be applied to the terms specified in terms. This vector must be of length one or the same length as terms. If it is of length one then the same constraint is applied to all the terms in terms. If any of the constraints are equal to NA then they are left unchanged for those terms.

initial.values — a character vector specifying the initial values for the terms specified in terms. This vector must be of length one or the same length as terms. If it is of length one then the same initial value is applied to all the terms in terms. If any of the initial.values are equal to NA then they are left unchanged for those terms.

... — further arguments passed to asreml and to wald.asreml.

Value

An asrtests object, which is a list containing:

1. asreml.obj: an asreml object containing the fit of the model after all boundary and singular terms have been removed;
2. wald.tab: a 4-column data.frame containing a pseudo-anova table for the fixed terms produced by wald.asreml;
3. test.summary: a data.frame with columns term, DF, denDF, p and action. A row is added to it for each term that is dropped, added or tested or a note that several terms have been added or removed. A row contains the name of the term, the DF, the p-value and the action taken. Possible codes are: Dropped, Retained, Swapped, Unswapped, Significant, Nonsignificant, Absent, Added, Removed and Boundary. If the changed model did not converge, Unconverged will be added to the code. Note that the logical asreml.obj$converge also reflects whether there is convergence.

See Also

asrtests, rmboundary.asrtests, testranfix.asrtests, testrcov.asrtests, newfit.asreml, sig.devn.reparam.asrtests, choose.model.asrtests

Examples

```
## Not run:
  terms <- "(Date/(Sources * (Type + Species)))"
  current.asrt <- addrm.terms.asrtests(terms, current.asrt, add = TRUE)

  current.asrt <- addrm.terms.asrtests("A + B", current.asrt, denDF = "algebraic")

## End(Not run)
```

---

alldiffs *Forms an object of S3-class* alldiffs *that stores the predictions for a model fitted using asreml, along with supplied statistics for all pairwise differences.*

Description

Creates an object of S3-class alldiffs that consists of a list containing the following components: predictions, differences, p.differences, sed, LSD and backtransforms. Predictions must be supplied to the functions while the others will be set only if they are supplies; those not supplied are set to NULL. It also has attributes response, response.title, term, classify and tdf, which will be set to the values supplied or NULL if none are supplied.

Usage

```
alldiffs(predictions, differences = NULL, p.differences = NULL,
         sed = NULL, LSD = NULL, backtransforms = NULL,
         response = NULL, response.title = NULL,
         term = NULL, classify = NULL, tdf = NULL)
```

Arguments

predictions	a data.frame containing the predicted values that is consistent with an object of class asremlPredict such as is stored in the pvals component of the prediction component of the value produced by predict.asreml. That is, in addition to variables classifying the predictions, it will include columns named predicted.value, standard.error and est.status; each row contains a single predicted value. It may also contain columns for the lower and upper confidence limits for the predictions. If LSD is not NULL, the mean LSD will be added as an attribute named meanLSD.
differences	a matrix containing all pairwise differences between the predictions; it should have the same number of rows and columns as there are rows in predictions.

`p.differences` a `matrix` containing p-values for all pairwise differences between the predictions; each p-value is computed as the probability of a t-statistic as large as or larger than the observed difference divided by its standard error. The degrees of freedom of the t distribution for computing it are computed as the denominator degrees of freedom of the F value for the fixed term, if available; otherwise, the degrees of freedom stored in the attribute `tdf` are used; the matrix should be of the same size as that for `differences`.

`sed` a `matrix` containing the standard errors of all pairwise differences between the predictions; they are used in computing the p-values.

`LSD` a `data.frame` containing the mean, minimum and maximum LSD for determining the significance of pairwise differences.

`backtransforms` a `data.frame` containing the backtransformed values of the predicted values that is consistent with an object of class `asremlPredict` such as is stored in the `pvals` component of the `prediction` component of the value produced by `predict.asreml`. That is, in addition to variables classifying the predictions, it will include columns named `backtransformed.predictions` and `est.status`; it may also contain columns for the lower and upper confidence limits; each row contains a single predicted value.

`response` a `character` specifying the response variable for the predictions. It is stored as an attribute to the `alldiffs` object.

`response.title` a `character` specifying the title for the response variable for the predictions. It is stored as an attribute to the `alldiffs` object.

`term` a character string giving the variables that define the term that was fitted using `asreml` and that corresponds to `classify`. It is often the same as `classify`. It is stored as an attribute to the `alldiffs` object.

`classify` a character string giving the variables that define the margins of the multiway table used in the prediction. Multiway tables are specified by forming an interaction type term from the classifying variables, that is, separating the variable names with the `:` operator. It is stored as an attribute to the `alldiffs` object.

`tdf` an `integer` specifying the degrees of freedom of the standard error. It is used as the degrees of freedom for the t-distribution on which p-values and confidence intervals are based. It is stored as an attribute to the `alldiffs` object.

Value

An object of S3-class `alldiffs` with attributes `response`, `response.title`, `term`, `classify` and `tdf`.

See Also

`asremlPlus-package`, `print.alldiffs`, `predictparallel.asreml`,`predictionplot.asreml`, `predictiondiffs.asreml`, `pred.present.asreml`

Examples

```
## Not run:
  Var.pred <- predict(current.asr, classify="Variety", sed=TRUE)$predictions
```

```
  wald.tab <-  current.asrt$wald.tab
  den.df <- wald.tab[match("Variety", rownames(wald.tab)), "denDF"]
  Var.diffs <- alldiffs(predictions = Var.pred$pvals,
                        sed = Var.pred$sed,
                        tdf = den.df)
  Var.diffs <- predictiondiffs.asreml(classify = "Variety",
                                      alldiffs.obj=Var.diffs)
  print.alldiffs(Var.diffs, which="differences")

## End(Not run)
```

`angular` *Applies the angular transformation to proportions.*

Description

Applies the angular transformation to numeric values. It is given by $\sin^{-1}(\sqrt{proportions})$

Usage

```
angular(proportions, n)
```

Arguments

`proportions`	The proportions.
`n`	The divisor(s) for each proportion

Value

A numeric.

See Also

`angular.mod`, `power.transform`.

Examples

```
n <-25
y <- rbinom(10, n, 0.5)
y <- c(y,0,n)
p <- y/n
p.ang <- angular(p, n)
```

angular.mod *Applies the modified angular transformation to a vector of counts.*

Description

Applies the angular transformation to a vector of counts. A modified transformation is used that is appropriate when N < 50 and the proportion is not between 0.3 and 0.7. The transformation is given by $\sin^{-1}\frac{count+0.375}{n+0.75}$ arcsin(sqrt((count + 0.375) / (n + 0.75))).

Usage

```
angular.mod(count, n)
```

Arguments

count	The numeric vector of counts.
n	The number(s) of observations from which the count(s) were obtained.

Value

A numeric vector.

See Also

angular, power.transform.

Examples

```
n <-25
y <- rbinom(10, n, 0.5)
y <- c(y,0,n)
p.ang.mod <- angular.mod(y, n)
```

asremlPlus-deprecated *Deprecated Functions in Package asremlPlus*

Description

These functions have been renamed and deprecated in asremlPlus: addrm.terms.asreml replaced by addrm.terms.asrtests, choose.model.asreml by choose.model.asrtests, recalc.wald.tab.asreml by recalc.wald.tab.asrtests, rmboundary.asreml by rmboundary.asrtests, sig.devn.reparam.asreml by sig.devn.reparam.asrtests, testranfix.asreml by testranfix.asrtests, testrcov.asreml by testrcov.asrtests, testswapran.asreml by testswapran.asrtests, info.crit by info.crit.asreml, reml.lrt by reml.lrt.asreml

Usage

```
addrm.terms.asreml(...)
choose.model.asreml(...)
recalc.wald.tab.asreml(...)
rmboundary.asreml(...)
sig.devn.reparam.asreml(...)
testranfix.asreml(...)
testrcov.asreml(...)
testswapran.asreml(...)
info.crit(...)
reml.lrt(...)
```

Arguments

...	absorbs arguments passed from the old functions of the style foo.bar().

Author(s)

Chris Brien

---

asrtests *Forms an object of S3-class* asrtests *that stores a fitted asreml object, a pseudo-anova table for the fixed and a history of changes and hypthesis testing used in obtaining the model.*

Description

An object of S3-class asrtests consists of a list containing:

1. asreml.obj: an asreml object containing the fit of the model;
2. wald.tab: a data.frame containing a pseudo-anova table for the fixed terms produced by wald.asreml, which will be called if wald.tab is NULL;
3. test.summary: a data.frame with columns term, DF, denDF, p and action. A row is added to it for each term that is dropped, added or tested or a note that several terms have been added or removed. A row contains the name of the term, the DF, the p-value and the action taken. Possible codes are: Dropped, Retained, Swapped, Unswapped, Significant, Nonsignificant, Absent, Added, Removed and Boundary. If the changed model did not converge, Unconverged will be added to the code. Note that the logical asreml.obj$converge also reflects whether there is convergence.

A call to asrtests with test.summary = NULL re-initializes the test.summary data.frame.

Usage

```
asrtests(asreml.obj, wald.tab = NULL, test.summary = NULL,
                denDF = "default", ...)
```

Arguments

asreml.obj	an asreml object for a fitted model.
wald.tab	a data.frame containing a pseudo-anova table for the fixed terms produced by wald.asreml; it should have 4 columns. Sometimes wald.asreml returns a data.frame and at other times a list. For example, it may return a list when denDF is used. In this case, the Wald component of the list is to be extracted and stored.
test.summary	a data.frame with columns term, DF, denDF, p and action containing the results of previous hypothesis tests.
denDF	Specifies the enthod to use in computing approximate denominator degrees of freedom when wald.asreml is called. Can be none to suppress the computations, numeric for numerical methods, algebraic for algebraic methods or default, the default, to autommatically choose numeric or algebraic computations depending on problem size. The denominator degrees of freedom are calculated according to Kenward and Roger (1997) for fixed terms in the dense part of the model.
...	further arguments passed to wald.asreml.

Value

An object of S3-class asrtests.

See Also

asremlPlus-package, testranfix.asrtests, choose.model.asrtests, rmboundary.asrtests, sig.devn.reparam.asrtests

Examples

```
## Not run:
data(Wheat.dat)

# Fit initial model
current.asr <- asreml(yield ~ Rep + WithinColPairs + Variety,
                      random = ~ Row + Column + units,
                      rcov = ~ ar1(Row):ar1(Column),
                      data=Wheat.dat)

# Load current fit into an asrtests object
current.asrt <- asrtests(current.asr, NULL, NULL)

# Check for and remove any boundary terms
current.asrt <- rmboundary.asrtests(current.asrt)

## End(Not run)
```

choose.model.asrtests *Determines the set of significant terms taking into account the hierarchy or marginality relations.*

Description

Performs a series of hypothesis tests taking into account the marginality of terms. In particular, a term will not be tested if it is marginal to (or nested in) one that is significant. For example, if A:B is significant, then neither A nor B will be tested. For a random term, the term is removed from the model fit, any boundary terms are removed using rmboundary.asrtests and a REML likelihood ratio test is performed using reml.lrt.asreml. If it is not significant and drop.ran.ns is TRUE, the term is permanently removed from the model. Note that if boundary terms are removed, the reduced model may not be nested in the full model in which case the test is not valid. For fixed terms, the Wald tests are performed and the p-value for the term obtained. If it is not significant and drop.fix.ns is TRUE, the term is permanently removed from the model. A row is added to test.summary for each term that is tested.

Usage

```
choose.model.asrtests(terms.marginality=NULL, asrtests.obj,
                      alpha = 0.05, drop.ran.ns=TRUE,
                      positive.zero = FALSE, bound.test.parameters = "none",
                      drop.fix.ns=FALSE, denDF = "default",  dDF.na = "none",
                      dDF.values = NULL, trace = FALSE, update = TRUE,
                      set.terms = NULL, ignore.suffices = TRUE,
                      constraints = "P", initial.values = NA, ...)
```

Arguments

terms.marginality
: a square matrix of ones and zeros with row and column names being the names of the terms. The diagonal elements should be one, indicating that a term is marginal to itself. Elements should be one if the row term is marginal to the column term. All other elements should be zero.

asrtests.obj
: an asrtests object for a fitted model that is a list containing an asreml object, a wald.tab data.frame with 4 columns, and a data.frame with 5 columns that records any previous changes and tests in the fitted model.

alpha
: the significance level for the test.

drop.ran.ns
: a logical indicating whether to drop nonsignificant random terms from the model.

positive.zero
: Indicates whether the hypothesized values for the variance components being tested are on the boundary of the parameter space. For example, this is true for positively-constrained variance components that, under the reduced model, are zero. This argument does not need to be set if bound.test.parameters is set.

bound.test.parameters
: Indicates whether for the variance components being tested, at least some of the hypothesized values are on the boundary of the parameter space. The default is

"none". Other possibilities are "onlybound" and "one-and-one". The latter signifies that there are two parameters being tested, one of which is bound and the other is not. For example, the latter is true for testing a covariance and a positively-constrained variance component that, under the reduced model, are zero.

drop.fix.ns — a logical indicating whether to drop a fixed term from the model when it is nonsignificant

denDF — Specifies the enthod to use in computing approximate denominator degrees of freedom when wald.asreml is called. Can be none to suppress the computations, numeric for numerical methods, algebraic for algebraic methods or default, the default, to autommatically choose numeric or algebraic computations depending on problem size. The denominator degrees of freedom are calculated according to Kenward and Roger (1997) for fixed terms in the dense part of the model.

dDF.na — the method to use to obtain substitute denominator degrees of freedom. when the numeric or algebraic methods produce an NA. If dDF.na = "none", no subtitute denominator degrees of freedom are employed; if dDF.na = "residual", the residual degrees of freedom from asreml.obj$nedf are used; if dDF.na = "maximum", the maximum of those denDF that are available, excluding that for the Intercept, is used; if all denDF are NA, asreml.obj$nedf is used. If dDF.na = "supplied", a vector of values for the denominator degrees of freedom is to be supplied in dDF.values. Any other setting is ignored and a warning message produced. Generally, substituting these degrees of freedom is anticonservative in that it is likely that the degrees of freedom used will be too large.

dDF.values — A vector of values to be used when dDF.na = "supplied". Its values will be used when denDF in a test for a fixed effect is NA. This vector must be the same length as the number of fixed terms, including (Intercept) whose value could be NA.

trace — if TRUE then partial iteration details are displayed when ASReml-R functions are invoked; if FALSE then no output is displayed.

update — if TRUE then update.asreml is called in testing models. In doing this the arguments R.param and G.param are set to those in the asreml object stored in asrtests.obj so that the values from the previous model are used as starting values. If FALSE then a call is made to asreml in which the only changes to the asreml.obj stored in the supplied asrtests.obj are (i) to the terms in the fixed and random models corresponding to terms in terms.marginality and (ii) those modifications specified via ....

set.terms — a character vector specifying the terms that are to have constraints and/or initial values set prior to fitting.

ignore.suffices — a logical vector specifying whether the suffices of the asreml-assigned names of the variance terms (i.e. the information to the right of an "!", other than "R!") is to be ignored in matching elements of terms. If TRUE for an element of terms, the suffices are stripped from the asreml-assigned names. If FALSE for an element of terms, the element must exactly match an asreml-assigned name for a variance term. This vector must be of length one or the same length

as terms. If it is of length one then the same action is applied to the asreml-assigned suffices for all the terms in terms.

constraints — a character vector specifying the constraints to be applied to the terms specified in terms. This vector must be of length one or the same length as terms. If it is of length one then the same constraint is applied to all the terms in terms. If any of the constraints are equal to NA then they are left unchanged for those terms.

initial.values — a character vector specifying the initial values for the terms specified in terms. This vector must be of length one or the same length as terms. If it is of length one then the same initial value is applied to all the terms in terms. If any of the initial.values are equal to NA then they are left unchanged for those terms.

... — further arguments passed to asreml and wald.asreml via testranfix.asrtests.

Value

A list containing:

1. asrtests.obj: an asrtests object, containing the asreml object correspondiing to the final fit, a wald.tab data.frame, and a test.summary data.frame that contains a record of the testing of the terms (see asrtests for more details);
2. sig.tests: a character vector whose elements are the the significant terms amongst those tested.

See Also

asrtests, testranfix.asrtests, testrcov.asrtests, reml.lrt.asreml, rmboundary.asrtests, newfit.asreml, addrm.terms.asrtests, sig.devn.reparam.asrtests

Examples

```
## Not run:
data(WaterRunoff.dat)
current.asr <- asreml(log.Turbidity ~ Benches + (Sources * (Type + Species)) * Date,
                      random = ~Benches:MainPlots:SubPlots:spl(xDay),
                      data = WaterRunoff.dat, keep.order = TRUE)
current.asrt <- asrtests(current.asr, NULL, NULL)
terms.treat <- c("Sources", "Type", "Species",
                 "Sources:Type", "Sources:Species")
terms <- sapply(terms.treat,
                FUN=function(term){paste("Date:",term,sep="")},
                simplify=TRUE)
terms <- c("Date", terms)
terms <- unname(terms)
marginality <-  matrix(c(1,0,0,0,0,0, 1,1,0,0,0,0,  1,0,1,0,0,0,
                         1,0,1,1,0,0, 1,1,1,0,1,0, 1,1,1,1,1,1), nrow=6)
rownames(marginality) <- terms
colnames(marginality) <- terms
choose <- choose.model.asrtests(marginality, current.asrt)
          current.asrt <- choose$asrtests.obj
          sig.terms <- choose$sig.terms

## End(Not run)
```

info.crit.asreml *Computes AIC and BIC for a model.*

Description

Computes Akiake and Bayesian (Schwarz) Information Criteria for a model. The function `info.crit` is provided for backwards compatibility.

Usage

```
info.crit.asreml(asreml.obj)
```

Arguments

`asreml.obj`	An `asreml` object resulting from the fitting of a model using REML.

Value

A data frame containing the Residual degrees of freedom, AIC, BIC and log of the REML value.

See Also

`reml.lrt.asreml`

Examples

```
## Not run:
   data(Wheat.dat)
   current.asr <- asreml(yield ~ Rep + WithinColPairs + Variety,
                         random = ~ Row + Column + units,
                         rcov = ~ ar1(Row):ar1(Column),
                         data=Wheat.dat)
    info.crit.asreml(current.asr)

## End(Not run)
```

newfit.asreml *refits an asreml model with modified model formula using either a call to* `update.asreml` *or a direct call to* `asreml`.

Description

Extracts the call from the `asreml.obj` and evaluates that call, replacing any arguments with changed values. If `update` is `TRUE` and `set.terms` is not set, the call is evaluated using `update.asreml`; otherwise, it is evaluated using a direct call to `asreml`. The principal difference is that the latter does not enforce the use of previous values of the variance parameters as initial values; it sets `G.param` and `R.param` to `NULL` or to values as specified for `set.terms`. The `...` argument can be used to pass `G.param` and/or `R.param`, provided `update` is `FALSE` and `set.terms` is not set.

Usage

```
newfit.asreml(asreml.obj, fixed., random., sparse., rcov.,
              update = TRUE, keep.order = TRUE, set.terms = NULL,
             ignore.suffices = TRUE, constraints = "P", initial.values = NA, ...)
```

Arguments

asreml.obj	a valid asreml object with with a component named call (from a previous call to either asreml or update.asreml).
fixed.	a character or formula specifying changes to the fixed formula. This is a two-sided formula where "." is substituted for existing components in the fixed component of asreml.obj$call.
random.	a character or formula specifying changes to the random formula. This is a one-sided formula where "." is substituted for existing components in the random component of asreml.obj$call.
sparse.	a character or formula specifying changes to the sparse formula. This is a one-sided formula where "." is substituted for existing components in the sparse component of asreml.obj$call.
rcov.	a character or formula specifying changes to the error formula. This is a one-sided formula where "." is substituted for existing components in the rcov component of asreml.obj$call.
update	a logical indicated whether to use update.asreml or asreml to evaluate the modified call. If TRUE, use update.asreml to evaluate the modified call. In doing this the arguments R.param and G.param are set to those in the asreml.obj so that the values from the previous model are used as starting values. If FALSE then a call is made to asreml itself, in which the only changes from the previous call are those specified in the arguments to newfit.asreml.
keep.order	a logical value indicating whether the terms should keep their positions. If FALSE the terms are reordered so that main effects come first, followed by the interactions, all second-order, all third-order and so on. Effects of a given order are kept in the order specified.
set.terms	a character vector specifying the terms that are to have constraints and/or initial values set prior to fitting.
ignore.suffices	a logical vector specifying whether the suffices of the asreml-assigned names of the variance terms (i.e. the information to the right of an "!", other than "R!") is to be ignored in matching elements of terms. If TRUE for an element of terms, the suffices are stripped from the asreml-assigned names. If FALSE for an element of terms, the element must exactly match an asreml-assigned name for a variance term. This vector must be of length one or the same length as terms. If it is of length one then the same action is applied to the asreml-assigned suffices for all the terms in terms.
constraints	a character vector specifying the constraints to be applied to the terms specified in terms. This vector must be of length one or the same length as terms. If it is of length one then the same constraint is applied to all the terms in terms. If any of the constraints are equal to NA then they are left unchanged for those terms.

initial.values	a character vector specifying the initial values for the terms specified in terms. This vector must be of length one or the same length as terms. If it is of length one then the same initial value is applied to all the terms in terms. If any of the initial.values are equal to NA then they are left unchanged for those terms.
...	additional arguments to the call, or arguments with changed values.

Value

An asreml object.

References

Butler, D. G., et al. (2010). *Analysis of Mixed Models for S language environments: ASReml-R reference manual.* Brisbane, DPI Publications.

See Also

update.asreml, setvarianceterms.asreml

Examples

```
## Not run:
    m2.asreml <- newfit.asreml(m1.asreml, random. = "~ . - Blocks:Plots", maxiter=75)

## End(Not run)
```

---

newrcov.asrtests *Fits a new rcov formula using* asreml.

Description

Fits a new rcov formula using asreml. Any boundary terms resulting after fitting the new rcov model are removed using rmboundary.asrtests. A row is added to the test.summary data.frame using the supplied label and the action Set.

Usage

```
newrcov.asrtests(terms=NULL, asrtests.obj, label = "R model",
                 update = TRUE, trace = FALSE, denDF="default",
                 set.terms = NULL, ignore.suffices = TRUE,
                 constraints = "P", initial.values = NA, ...)
```

Arguments

terms	a model for the rcov argument in asreml, stored as a character.
asrtests.obj	an asrtests object for a fitted model that is a list containing an asreml object, a wald.tab data.frame with 4 columns, and a data.frame with 5 columns that records any previous changes and tests in the fitted model.
label	a character string to use as the label in test.summary and which indicates what is being tested.
update	if TRUE then update.asreml is called to fit the model with the rcov model supplied in terms. In doing this the arguments R.param and G.param are set to those in the asreml object stored in asrtests.obj so that the values from the previous model are used as starting values. If FALSE then a call is made to asreml in which the only changes from the previous call are that (i) rcov model is that specified in terms and (ii) modifications specified via ... are made.
trace	if TRUE then partial iteration details are displayed when ASReml-R functions are invoked; if FALSE then no output is displayed.
denDF	Specifies the method to use in computing approximate denominator degrees of freedom when wald.asreml is called. Can be none to suppress the computations, numeric for numerical methods, algebraic for algebraic methods or default, the default, to autommatically choose numeric or algebraic computations depending on problem size. The denominator degrees of freedom are calculated according to Kenward and Roger (1997) for fixed terms in the dense part of the model.
set.terms	a character vector specifying the terms that are to have constraints and/or initial values set prior to fitting.
ignore.suffices	a logical vector specifying whether the suffices of the asreml-assigned names of the variance terms (i.e. the information to the right of an "!", other than "R!") is to be ignored in matching elements of terms. If TRUE for an element of terms, the suffices are stripped from the asreml-assigned names. If FALSE for an element of terms, the element must exactly match an asreml-assigned name for a variance term. This vector must be of length one or the same length as terms. If it is of length one then the same action is applied to the asreml-assigned suffices for all the terms in terms.
constraints	a character vector specifying the constraints to be applied to the terms specified in terms. This vector must be of length one or the same length as terms. If it is of length one then the same constraint is applied to all the terms in terms. If any of the constraints are equal to NA then they are left unchanged for those terms.
initial.values	a character vector specifying the initial values for the terms specified in terms. This vector must be of length one or the same length as terms. If it is of length one then the same initial value is applied to all the terms in terms. If any of the initial.values are equal to NA then they are left unchanged for those terms.
...	further arguments passed to asreml and to wald.asreml.

Value

An asrtests object, which is a list containing:

1. asreml.obj: an asreml object containing the fit after the term has been omitted from the model;
2. wald.tab: a 4-column data.frame containing a pseudo-anova table for the fixed terms produced by wald.asreml;
3. test.summary: a data.frame with columns term, DF, denDF, p and action. A row is added to it for each term that is dropped, added or tested or a note that several terms have been added or removed. A row contains the name of the term, the DF, the p-value and the action taken. Possible codes are: Dropped, Retained, Swapped, Unswapped, Significant, Nonsignificant, Absent, Added, Removed and Boundary. If the changed model did not converge, Unconverged will be added to the code. Note that the logical asreml.obj$converge also reflects whether there is convergence.

If the term is not in the model, then the supplied asreml object will be returned. Also, reml.test will have the likelihood ratio and the p-value set to NA and the degrees of freedom to zero. Similarly, the row of test.summary for the term will have its name, a p-value set to NA, and action set to Absent.

See Also

asremlPlus-package, asrtests, testrcov.asrtests,

choose.model.asrtests, reml.lrt.asreml, rmboundary.asrtests, newfit.asreml, testswapran.asrtests, addrm.terms.asrtests, sig.devn.reparam.asrtests

Examples

```
## Not run:
data(Wheat.dat)
current.asr <- asreml(yield ~ Rep + WithinColPairs + Variety,
                      random = ~ Row + Column + units,
                      rcov = ~ ar1(Row):ar1(Column),
                      data=Wheat.dat)
current.asrt <- asrtests(current.asr, NULL, NULL)
current.asrt <- rmboundary.asrtests(current.asrt)
# Fit Row autocorrelation
current.asrt <- newrcov.asrtests("~ Row:ar1(Column)", current.asrt,
                                 label="Row autocorrelation")
print(current.asrt)

## End(Not run)
```

---

num.recode *Recodes the unique values of a vector using the values in a new vector.*

Description

Recodes the unique values of a variate using the value in position i of the new.values vector to replace the ith sorted unique values of x. The new levels do not have to be unique.

Usage

```
num.recode(x, new.values)
```

Arguments

x	The vector to be recoded.
new.values	A vector of length unique(x) containing values to use in the recoding.

Value

A vector.

Author(s)

Chris Brien

See Also

dae::fac.recode.

Examples

```
## set up a factor with labels
x <- rep(c(-42, -14, 14, 42), 4)

## recode x
b <- num.recode(x, c(0, 28, 56, 84))
```

---

permute.square *Permutes the rows and columns of a square matrix.*

Description

Permutes the rows and columns of a square matrix.

Usage

```
permute.square(x, permutation)
```

Arguments

x	A square matrix.
permutation	A vector specifying the new order of rows and columns.

Value

A square matrix.

See Also

permute.to.zero.lowertri

Examples

```
terms.marginality <-  matrix(c(1,0,0,0,0,  0,1,0,0,0, 0,1,1,0,0,
                               1,1,1,1,0, 1,1,1,1,1), nrow=5)
permtn <- c(1,3,2,4,5)
terms.marginality <- permute.square(terms.marginality, permtn)
```

`permute.to.zero.lowertri` *Permutes a square matrix until all the lower triangular elements are zero.*

Description

Permutes a square matrix until all the lower triangular elements are zero.

Usage

```
permute.to.zero.lowertri(x)
```

Arguments

x	A square matrix of order n with at least n*(n-1)/2 zero elements.

Value

A square `matrix`.

See Also

permute.square

Examples

```
terms.marginality <-  matrix(c(1,0,0,0,0,  0,1,0,0,0, 0,1,1,0,0,
                               1,1,1,1,0, 1,1,1,1,1), nrow=5)
terms.marginality <- permute.to.zero.lowertri(terms.marginality)
```

`plotvariofaces.asreml` *plot empirical variogram faces, including envelopes, from supplied residuals as described by Stefanova, Smith & Cullis (2009)*

Description

Produces a plot for each face of an empirical 2D `variogram` based on supplied residuals from both an observed data set and simulated data sets. Those from simulated data sets are used to produce confidence envelopes If the data consists of sections, such as separate experiments, the two variogram faces are produced for each section. This function is less efficient in storage terms than `variofaces.asreml`, because here the residuals from all simulated data sets must be saved, in addition to the values for the variogram faces; in `variofaces.asreml`, the residuals for each simulated data set are discarded after the variogram has been calculated. On the other hand, the present function is more flexible, because there is no restriction on how the residuals are obtained.

Usage

```
plotvariofaces.asreml(data, residuals, restype="Residuals", ...)
```

Arguments

`data`	A `data.frame` with either 3 or 4 columns. Only if there are 4 columns, the first should be a factor indexing sections for which separate variogram plots are to be produced. In either case, the other 3 columns should be, in order, (i) a factor indexing the x-direction, (ii) a factor indexing the y-direction, and (iii) the residuals for the observed response.
`residuals`	A `data.frame`, with with either 2 or 3 initial columns followed by columns, each of which are the residuals from a simulated data set.
`restype`	A `character` describing the type of residuals that have been supplied. It will be used in the plot titles.
`...`	Other arguments that are passed down to the function `asreml.variogram`.

Details

For each set of residuals, `asreml.variogram` is used to obtain the empirical variogram, from which the values for its faces are obtained. Plots are produced for each face and include the observed residuals and the 2.5%, 50% & 97.5% quantiles.

Value

A `list` with the following components:

1. **face1:** a `data.frame` containing the variogram values on which the plot for the first dimension is based.
2. **face2:** a `data.frame` containing the variogram values on which the plot for the second dimension is based.

Author(s)

Chris Brien

References

Stefanova, K. T., Smith, A. B. & Cullis, B. R. (2009) Enhanced diagnostics for the spatial analysis of field trials. *Journal of Agricultural, Biological, and Environmental Statistics*, **14**, 392–410.

See Also

asremlPlus-package, asreml, asreml.variogram, variofaces.asreml, simulate.asreml.

Examples

```
## Not run:
data(Wheat.dat)
current.asr <- asreml(yield ~ Rep + WithinColPairs + Variety,
                      random = ~ Row + Column + units,
                      rcov = ~ ar1(Row):ar1(Column),
                      data=Wheat.dat)
current.asrt <- asrtests(current.asr, NULL, NULL)
current.asrt <- rmboundary.asrtests(current.asrt)
# Form variance matrix based on estimated variance parameters
s2 <- current.asr$sigma2
gamma.Row <- current.asr$gammas[1]
gamma.unit <- current.asr$gammas[2]
rho.r <- current.asr$gammas[4]
rho.c <- current.asr$gammas[5]
row.ar1 <- mat.ar1(order=10, rho=rho.r)
col.ar1 <- mat.ar1(order=15, rho=rho.c)
V <- gamma.Row * fac.sumop(Wheat.dat$Row) +
  gamma.unit * diag(1, nrow=150, ncol=150) +
  mat.dirprod(col.ar1, row.ar1)
V <- s2*V

#Produce variogram faces plot (Stefanaova et al, 2009)
resid <- simulate.asreml(current.asr, V=V, which="residuals")
resid$residuals <- cbind(resid$observed[c("Row","Column")],
                         resid$residuals)
plotvariofaces.asreml(data=resid$observed[c("Row","Column","residuals")],
                      residuals=resid$residuals,
                      restype="Standardized conditional residuals")

## End(Not run)
```

`power.transform` *Perform a combination of a linear and power transformation on a variable whose name is given as a character string in* `var.name`. *The transformed variable is stored in the* `data.frame` `data`.

Description

Perform a combination of a linear and a power transformation on a variable whose name is given as a character string in `var.name`. The transformed variable is stored in the `data.frame data`. The name of the transformed variable is made by prepending to the original `var.name` a combination of (i) `.offset`, if `offset` is nonzero, (ii) `neg.`, if `scale` is -1, or `scaled.`, if `abs(scale)` is other than one, and (iii) either `log.`, `sqrt.`, `recip.` or `power.`, if `power` is other than one. No action is taken if there is no transformation (i.e. `offset = 0`, `scale = 1` and `power = 1`). Also, the `titles` list is extended to include a component with a generated title for the transformed variable with text indicating the transformation prepended to the `title` for the `var.name` obtained from the `titles list`. For nonzero `offset`, 'Offset ' is prepended, For `scaled` not equal to one, the possible prepends are 'Negative of ' and 'Scaled '. The possible prepended texts for `power` not equal to one are 'Logarithm of', 'Square root of ', 'Reciprocal of ' and 'Power nnnn of ', where `nnn` is the power used.

Usage

```
power.transform(var.name, power = 1, offset = 0, scale = 1, titles = NULL, data)
```

Arguments

`var.name`	A `character string` specifying the name of the variable in the `data.frame data` that is to be transformed.
`power`	A number specifying the power to be used in the transformation. If equal to 1, the default, no power transformation is applied. Otherwise, the variable is raised to the specified `power`, after scaling and applying any nonzero `offset`. If `power = 0`, the natural logarithm is used to transform the response; however, if the smallest value to be log-transformed is less than 1e-04, an error is generated. A log-transformation in this situation may be possible if a nonzero `offset` and/or a `scale` not equal to one is used.
`offset`	A number to be added to each value of the variable, after any scaling and before applying any power transformation.
`scale`	A number to multiply each value of the variable, before adding any offset and applying any power transformation.
`titles`	A `character vector`, each element of which is named for a variable in `data` and is a `character string` giving a title to use in output (e.g. tables and graphs) involving the variable. If titles are not supplied, the column name of the variable in `dta` is used.
`data`	A `data.frame` containing the variable to be transformed and to which the transformed variable is to be appended.

Value

A `list` with a component named `data` that is the `data.frame` containing the transformed variable, a component named tvar.name that is a `character string` that is the name of the transformed variable in `data`, and a component named `titles` that extends the `list` supplied in the `titles` argument to include a generated title for the transformed title, the name of the new component being `tvar.name`.

Author(s)

Chris Brien

See Also

angular, angular.mod.

Examples

```
## set up a factor with labels
x.dat <- data.frame(y = c(14, 42, 120, 150))

## transform y to logarithms
trans <- power.transform("y", power = 0, titles=list(y = "Length (cm)"), data = x.dat)
x.dat <- trans$data
tvar.name <- trans$tvar.name

## transform y to logarithms after multiplying by -1 and adding 1.
z.dat <- data.frame( y = c(-5.25, -4.29, -1.22, 0.05))
trans <- power.transform("y", power = 0, scale = -1, offset = 1 ,
                         titles=list(y = "Potential"), data = z.dat)
z.dat <- trans$data
tvar.name <- trans$tvar.name
```

---

pred.present.asreml *This function forms the predictions for each significant term and presents them in tables and/or graphs.*

Description

This function forms the predictions for each term in terms using a supplied asreml object and predictparallel.asreml. Tables are produced using predictparallel.asreml, in conjunction with predictiondiffs.asreml, with the argument tables specifying which tables are printed. The argument plots, along with transform.power, controls which plots are produced. The plots are produced using predictionplot.asreml, with line plots produced when variables involving x.num or x.fac are involved in classify for the predictions and bar charts otherwise. In order to get the correct predictions you may need to supply additional arguments to predict through ... e.g. present.

Usage

```
pred.present.asreml(terms, asreml.obj = NULL,
                    wald.tab = NULL, dDF.na = "residual", dDF.values = NULL,
                    x.num = NULL, x.fac = NULL, nonx.fac.order = NULL,
                    x.pred.values = NULL, x.plot.values = NULL,
                    plots = "predictions", panels = "multiple",
                    graphics.device = NULL,
                    error.intervals = "Confidence", avsed.tolerance = 0.25,
```

```
                titles = NULL, colour.scheme = "colour", save.plots = FALSE,
                transform.power = 1, offset = 0, scale = 1,
                pairwise = TRUE, tables = "all", levels.length = NA,
                alpha = 0.05, inestimable.rm = TRUE,
                trace = FALSE, ggplotFuncs = NULL, ...)
```

Arguments

terms	a character vector giving the terms for which predictions are required.
asreml.obj	asreml object for a fitted model.
wald.tab	a data frame containing the pseudo-anova table for the fixed terms produced by a call to wald.asreml. The main use of it here is in getting denominator degrees of freedom when confidence intervals are to be plotted.
dDF.na	the method to use to obtain approximate denominator degrees of freedom. when the numeric or algebraic methods produce an NA. Consistent with when no denDF are available, the default is "residual" and so the residual degrees of freedom from asreml.obj$nedf are used. If dDF.na = "none", no subtitute denominator degrees of freedom are employed; if dDF.na = "maximum", the maximum of those denDF that are available, excluding that for the Intercept, is used; if all denDF are NA, asreml.obj$nedf is used. If dDF.na = "supplied", a vector of values for the denominator degrees of freedom is to be supplied in dDF.values. Any other setting is ignored and a warning message produced. Generally, substituting these degrees of freedom is anticonservative in that it is likely that the degrees of freedom used will be too large.
dDF.values	A vector of values to be used when dDF.na = "supplied". Its values will be used when denDF in a test for a fixed effect is NA. This vector must be the same length as the number of fixed terms, including (Intercept) whose value could be NA.
x.num	A character string giving the name of the numeric covariate that corresponds to x.fac, is potentially included in terms in the fitted model and which corresponds to the x-axis variable. It should have the same number of unique values as the number of levels in x.fac.
x.fac	A character string giving the name of the factor that corresponds to x.num, is potentially included in terms in the fitted model and which corresponds to the x-axis variable. It should have the same number of levels as the number of unique values in x.num. The levels of x.fac must be in the order in which they are to be plotted - if they are dates, then they should be in the form yyyymmdd which can be achieved using as.Date.
nonx.fac.order	A character vector giving the order in which factors other than x.fac are to be plotted in plots with multiple panels (i.e. where the number of non-x factors is greater than 1). The first factor in the vector will be plotted on the X axis (if there is no x.num or x.fac. Otherwise, the order of plotting the factors is in columns (X facets) and then rows (Y facets). By default the order is in decreasing order for the numbers of levels of the non x factors.
x.pred.values	The values of x.num for which predicted values are required.

`x.plot.values` The actual values to be plotted on the x axis or in the labels of tables. They are needed when values different to those in `x.num` are to be plotted or `x.fac` is to be plotted because there is no `x.num` term corresponding to the same term with `x.fac`.

`plots` Possible values are `"none"`, `"predictions"`, `"backtransforms"` and `"both"`. Plots are not produced if the value is `"none"`. If data are not transformed for analysis (`transform.power = 1`), a plot of the predictions is produced provided `plots` is not `"none"`. If the data are transformed, the value of `plots` determines what is produced.

`panels` Possible values are `"single"` and `"multiple"`. When line plots are to be produced, because variables involving `x.num` or `x.fac` are involved in `classify` for the predictions, `panels` determines whether or not a single panel or multiple panels in a single window are produced. The `panels` argument is ignored for for bar charts.

`graphics.device`

A `character` specifying a graphics device for plotting. The default is `graphics.device = NULL`, which will result in plots being produced on the current graphics device. Setting it to `"windows"`, for example, will result in a windows graphics device being opened.

`error.intervals`

A character string indicating the type of error interval, if any, to calculate and plot in order to indicate uncertainty in the results. Possible values are `"none"`, `"StandardError"`, `"Confidence"` and `"halfLeastSignificant"`. The default is for confidence limits to be used. The `"halfLeastSignificant"` option results in half the Least Significant Difference (LSD) being added and subtracted to the predictions, the LSD being calculated using the average of the standard errors of all pairwise differences (SEDs) between the predictions. However, if the range of the SEDs divided by the average of the SEDs exceeds `avsed.tolerance`, calculations and plotting will revert to confidence intervals. Also, half LSDs cannot be used for backtansformed values and so confidence intervals will be used instead.

`avsed.tolerance`

The values of the range of the SEDs divided by the average of the SEDs that, if exceeded, will cause calculations and plotting to revert to confidence intervals. It should be a value between 0 and 1.

`titles` A `list`, each component of which is named for an object name and contains a `character string` giving a title to use in output (e.g. tables and graphs) for the object. Here they will be used for axis labels.

`colour.scheme` A character string specifying the colour scheme for the plots. The default is `"colour"` which produces coloured lines and bars, a grey background and white gridlines. A value of `"black"` results in black lines, grey bars and gridlines and a white background.

`save.plots` A `logical` that determines whether any plots will be saved. If they are to be saved, a file name will be generated that consists of the following elements separated by full stops: the response variable name with `.back` if backtransformed values are being plotted, the classify term, `Bar` or `Line` and, if `error.intervals` is not `"none"`, one of `SE`, `CI` or `LSI`. The file will be saved as a 'png' file in the current work directory.

`transform.power`
: A number specifying the power of a transformation, if one has been applied to the response variable. Unless it is equal to 1, the default, back-transforms of the predictions will be obtained and presented in tables or graphs as appropriate. The back-transform will raise the predictions to the power equal to the reciprocal of `transform.power`, unless it equals 0 in which case the exponential will be taken. Any scaling and offsetting will also be taken into account in the backtransformation.

`offset`
: A number that has been added to each value of the response after any scaling and before applying any power transformation. Unless it is equal to 0, the default, back-transforms of the predictions will be obtained and presented in tables or graphs as appropriate. The back-transform will, after backtransforming for any power transformation and scaling for any scale transformation, subtract the `offset`.

`scale`
: A number by which each value of the response has been multiply before adding any offset and applying any power transformation. Unless it is equal to 1, the default, back-transforms of the predictions will be obtained and presented in tables or graphs as appropriate. The back-transform will, after backtransforming for any power transformation, divide by the `scale`.

`pairwise`
: A logical indicating whether all pairise differences of the `predictions` and their standard errors and p-values are to be computed and stored. If `tables` is equal to `"differences"` or `"all"` or `error.intervals` is equal to `"halfLeastSignificant"`, they will be stored irrespective of the value of `pairwise`.

`tables`
: The elements of the `alldiffs` object to print. Possible values are `"none"`, `"predictions"`, `"backtransforms"`, `"nodifferences"`, `"differences"` and `"all"`. The option `"nodifferences"` will result in `"predictions"` and `"backtransforms"` being printed. The option `"differences"` will also result in the printing of the `predictions` from the `alldiffs` object.

`levels.length`
: The maximum number of characters from the the levels of factors to use in the row and column labels of the tables produced by `predictiondiffs.asreml`.

`alpha`
: the significance level for the test or 1 - alpha is the confidence level for confidence intervals.

`inestimable.rm`
: A `logical` indicating whether rows for predictions that are not estimable are to be removed from the components of the `alldiffs` object.

`trace`
: if TRUE then partial iteration details are displayed when ASReml-R functions are invoked; if FALSE then no output is displayed.

`ggplotFuncs`
: A `list`, each element of which contains the results of evaluating a `ggplot` function. It is created by calling the `list` function with a `ggplot` function call for each element. It is passed to `predictionplot.asreml`.

`...`
: further arguments passed to `predict.asreml` via `predictparallel.asreml` and to `ggplot` via `predictionplot.asreml`.

Value

a list containing a `alldiffs` object for each `term` for which tables are produced. The names of the components of this list are the `terms` with full-stops (`.`) replacing colons (`:`).

See Also

predictparallel.asreml, predictiondiffs.asreml, predictionplot.asreml, print.alldiffs, as.Date, Devices

Examples

```
## Not run:
data(WaterRunoff.dat)
titles <- list("Days since first observation", "Days since first observation",
               "pH", "Turbidity (NTU)")
names(titles) <- names(WaterRunoff.dat)[c(5,7,11:12)]
current.asr <- asreml(fixed = log.Turbidity ~ Benches + Sources + Type + Species +
                                Sources:Type + Sources:Species + Sources:Species:xDay +
                                Sources:Species:Date,
                      data = WaterRunoff.dat, keep.order = TRUE)
current.asrt <- asrtests(current.asr, NULL, NULL)
diff.list <- pred.present.asreml("Date:Sources:Species",
                                asreml.obj = current.asrt$asreml.obj,
                                wald.tab = current.asrt$wald.tab,
                                x.num = "xDay", x.fac = "Date",
                                x.pred.values=sort(unique(WaterRunoff.dat$xDay)),
                                x.plot.values=sort(unique(WaterRunoff.dat$Day)),
                                plots = "predictions",
                                error.intervals = "StandardError",
                                titles = titles,
                                transform.power = 0,
                                present = c("Type","Species","Sources"),
                                tables = "differences", levels.length = 6)

## End(Not run)
```

`predictiondiffs.asreml`

Uses information in a supplied alldiffs object to forms all pairwise differences between a set of predictions, the p-values for a test of whether the differences are significantly different from zero, and the minimum, mean and maximum LSD values, provided they are not present in the suppplied alldiffs object.

Description

Uses predictions and standard errors of pairwise differences from an alldiffs object to form, for those components not already present, (i) a table of all pairwise differences of the predictions in an alldiffs object, (ii) the p-values of each pairwise difference, and (iii) the minimum, mean and maximum LSD values. Predictions that are aliased (or nonestimable) are removed from the predictions component of the alldiffs object and standarard errors of differences involving them are removed from the sed component.

Each p-value is computed as the probability of a t-statistic as large as or larger than the absolute value of the observed difference divided by its standard error. The p-values are stored in the p.differences component. The degrees of freedom of the t-distribution is the degrees of freedom stored in the tdf attribute of the alldiffs object. This t-distibrution is also used in calculating the LSD statistics stored in the alldiffs object.

Usage

```
predictiondiffs.asreml(classify, alldiffs.obj,
                       x.num = NULL, x.fac = NULL,
                       levels.length = NA,
                       pairwise = TRUE, alpha = 0.05,
                       inestimable.rm = TRUE)
```

Arguments

classify	a character string giving the variables that define the margins of the multiway table to be predicted. Multiway tables are specified by forming an interaction type term from the classifying variables, that is, separating the variable names with the : operator.
alldiffs.obj	An alldiffs object for a fitted model. Note that the attribute tdf, being the degrees of freedom for the critical t-value to be used incomputing p-values, should be set to an appropriate value.
x.num	A character string giving the name of the numeric covariate that corresponds to x.fac, is potentially included in terms in the fitted model and which corresponds to the x-axis variable. It should have the same number of unique values as the number of levels in x.fac.
x.fac	A character string giving the name of the factor that corresponds to x.num, is potentially included in terms in the fitted model and which corresponds to the x-axis variable. It should have the same number of levels as the number of unique values in x.num. The levels of x.fac must be in the order in which they are to be plotted - if they are dates, then they should be in the form yyyymmdd which can be achieved using as.Date.
levels.length	The maximum number of characters from the the levels of factors to use in the row and column labels of the tables of pairwise differences and their p-values and standard errors.
pairwise	A logical indicating whether all pairwise differences of the predictions and their standard errors and p-values are to be computed and stored. If FALSE, the components differences and p.differences will be NULL in the returned alldiffs object.
alpha	The significance level for an LSD to compare a pair of predictions.
inestimable.rm	A logical indicating whether rows for predictions that are not estimable are to be removed from the components of the alldiffs object.

Value

An alldiffs object that is a list with components predictions containing the predictions and their standard errors, differences containing all pairwise differences between the predictions,

p.differences containing p-values for all pairwise differences between the predictions, sed containing the standard errors of all pairwise differences between the predictions, and,an LSD containing the mean, minimum and maximum LSDs.

See Also

asremlPlus-package, alldiffs, print.alldiffs, predictionplot.asreml, predictparallel.asreml, pred.present.asreml

Examples

```
## Not run:
  Var.pred <- predict(current.asr, classify="Variety", sed=TRUE)$predictions
  wald.tab <-  current.asrt$wald.tab
  den.df <- wald.tab[match("Variety", rownames(wald.tab)), "denDF"]
  Var.diffs <- alldiffs(predictions = Var.pred$pvals,
                        sed = Var.pred$sed,
                        tdf = den.df)
  Var.diffs <- predictiondiffs.asreml(classify = "Variety",
                                      alldiffs.obj=Var.diffs)
  print.alldiffs(Var.diffs, which="differences")

## End(Not run)
```

---

predictionplot.asreml *This function plots the predictions for a term, possibly with error bars.*

Description

This function plots the predictions y that are based on classify stored in the data.frame data. The package ggplot2 is used to produce the plots. Line plots are produced when variables involving x.num or x.fac are involved in classify for the predictions; otherwise, bar charts are produced. Further, for line charts, the argument panels determines whether a single plot or multiple plots in a single window are produced; for bar charts, the argument panels is ignored.

Usage

```
predictionplot.asreml(classify, y, data,
                      x.num = NULL, x.fac = NULL, nonx.fac.order = NULL,
                      colour.scheme = "colour", panels = "multiple",
                      graphics.device = NULL,
                      error.intervals = "Confidence",  titles = NULL,
                      y.title = NULL, filestem = NULL, ggplotFuncs = NULL, ...)
```

Arguments

classify	a character string giving the combinations of the independent variables on which the predictions are based. It is an interaction type term formed from the independent variables, that is, separating the variable names with the : operator.
y	a character string giving the name of the variable that is to be plotted on the Y axis.
data	a data.frame containing the values of the variables to be plotted. It should be consistent with an object of class asremlPredict such as is stored in the pvals component of the predictions component of the value produced by predict.asreml; that is, in addition to variables classifying the predictions, it will include a column with the name specified in the y argument, usually predicted.value or backtransformed.predictions; each row contains a single predicted value. If error.intervals is not "none", then the predictions component and, if present, the backtransforms component should contain columns for the lower and upper values of the limits for the interval with names that begin with lower and upper, respectively. The second part of the name must be one of Confidence, StandardError or halfLeastSignificant. The last part needs to be consistent between the lower and upper limits.
x.num	A character string giving the name of the numeric covariate that corresponds to x.fac, is potentially included in terms in the fitted model and which corresponds to the x-axis variable. It should have the same number of unique values as the number of levels in x.fac.
x.fac	A character string giving the name of the factor that corresponds to x.num, is potentially included in terms in the fitted model and which corresponds to the x-axis variable. It should have the same number of levels as the number of unique values in x.num. The levels of x.fac must be in the order in which they are to be plotted - if they are dates, then they should be in the form yyyymmdd which can be achieved using as.Date.
nonx.fac.order	A character vector giving the order in which factors other than x.fac are to be plotted in facetted plots (i.e. where the number of non x factors is greater than 1). The first factor in the vector will be plotted on the X axis (if there is no x.num or x.fac. Otherwise, the order of plotting the factors is in columns (X facets) and then rows (Y facets). By default the order is in decreasing order for the numbers of levels of the non x factors.
colour.scheme	A character string specifying the colour scheme for the plots. The default is "colour" which produces coloured lines and bars, a grey background and white gridlines. A value of "black" results in black lines, grey bars and gridlines and a white background.
panels	Possible values are "single" and "multiple". When line plots are to be produced, because variables involving x.num or x.fac are involved in classify for the predictions, panels determines whether or not a single panel or multiple panels in a single window are produced. The panels argument is ignored for bar charts.
graphics.device	A character specifying a graphics device for plotting. The default is graphics.device = NULL, which will result in plots being produced on the

current graphics device. Setting it to "windows", for example, will result in a windows graphics device being opened.

error.intervals
: A character string indicating the type of error interval, if any, to calculate and plot in order to indicate uncertainty in the results. Possible values are "none", "StandardError", "Confidence" and "halfLeastSignificant". Here, any option other than "none" will result in the interval limits contained in data being plotted.

titles
: A list, each component of which is named for an object name and contains a character string giving a title to use in output (e.g. tables and graphs) for the object. Here they will be used for axis labels for nonresponse variables. For response variable labels see y.title.

filestem
: A character sting giving the beginning of the name of the file in which to save the plot. If filestem = NULL, the plot is not saved. The remainder of the file name will be generated automatically and consists of the following elements separated by full stops: the classify term, Bar or Line and, if error.intervals is not "none", one of SE, CI or LSI. The file will be saved as a 'png' file in the current work directory.

y.title
: The title to be displayed on the y axis of any plot.

ggplotFuncs
: A list, each element of which contains the results of evaluating a ggplot function. It is created by calling the list function with a ggplot function call for each element.

...
: further arguments passed to ggplot.

Value

no values are returned.

See Also

predictiondiffs.asreml, pred.present.asreml, ggplot, Devices

Examples

```
## Not run:
data(WaterRunoff.dat)
current.asr <- asreml(fixed = log.Turbidity ~ Benches + Sources + Type + Species +
                              Sources:Type + Sources:Species +
                              Sources:xDay + Species:xDay + Species:Date,
                      data = WaterRunoff.dat, keep.order = TRUE)
current.asrt <- asrtests(current.asr, NULL, NULL)
predictions <- predict(current.asr, class="Species:Date:xDay",
                       present = c("Type","Species","Sources"),
                       levels=list(xDay=unique(WaterRunoff.dat$xDay)))$predictions$pvals
predictions <- predictions[predictions$est.status == "Estimable",]
predictionplot.asreml(classify="Species:Date:xDay", y = "predicted.value",
                      data = predictions, wald.tab = current.asrt$wald.tab,
                      x.num = "xDay", x.fac = "Date",
                      x.title = "Days since first observation",
```

```
                        y.title = "Predicted log(Turbidity)",
                        present = c("Type","Species","Sources"),
                        error.intervals = "none",
                        ggplotFuncs = list(ggtitle("Transformed turbidity over time")))

  diffs <- predictparallel.asreml(classify="Species:Date:xDay",
                                  present=c("Type","Species","Sources"),
                                  asreml.obj = current.asr, tables = "none",
                                  x.num = "xDay", x.fac = "Date",
                                  x.pred.values=sort(unique(WaterRunoff.dat$xDay)),
                                  x.plot.values=c(0,28,56,84),
                                  wald.tab = current.asrt$wald.tab)
  x.title <- "Days since first observation"
  names(x.title) <- "xDay"
  predictionplot.asreml(classify="Species:Date:xDay", y = "predicted.value",
                        data = diffs$predictions, wald.tab = current.asrt$wald.tab,
                        x.num = "xDay", x.fac = "Date",
                        titles = x.title,
                        y.title = "Predicted log(Turbidity)")

  ## End(Not run)
```

`predictparallel.asreml`

Uses an asreml object and a wald.tab to form the predictions and associated statistics for a term. It stores the results in an object of class alldifffs and may print the results. It can be used when a numeric vector and a factor that have parallel values both occur in the model and need to be taken into account.

Description

This function forms the predictions for `term` using `classify` and the supplied `asreml` object and stores them in an `alldiffs` object. If `x.num` is supplied, the predictions will be obtained for the values supplied in `x.pred.values` and, if supplied, `x.plot.values` will replace them in the `alldiffs` object that is returned. If `x.fac`, but not `x.num`, is specified, predictions will involve it and, if supplied, `x.plot.values` will replace the levels of `x.fac` in the `alldiffs` object that is returned. In order to get the correct predictions you may need to supply additional arguments to predict through `...` e.g. `present`. Any aliased predictions will be removed, as will any standard error of pairwise differences involving them.

Also calculated are the approximate degrees of freedom of the standard errors of the predictions. If the deominator degrees of freedom for `term` are available in `wald.tab`, they are used. Otherwise the residual degrees of freedom or the maximum of the denominator degrees in `wald.tab`, excluding the Intercept, are used. Which is used depends on the setting of `dDF.na`. These degrees of freedom are used for the t-distribution on which p-values and confidence intervals are based. It is stored as an attribute to the `alldiffs` object. The degrees of freedom are also used in valculating the minimum, mean and maximum LSD fro comparing pairs of predictions, which are also stored in the `alldiffs` object.

If pairwise = TRUE, all pairwise differences between the predictions, their standard errors, p-values and LSD statistics are computed using predictiondiffs.asreml. This adds them to the alldiffs object as additional list components named differences, sed, p.differences and LSD.

If a transformation has been applied (any one of transform.power is not one, scale is not one and offset is nonzero), the back-transforms of the predicted values and their lower and upper confidence intervals are added to a data.frame that is consistent with an object of class asremlPredict, such as is stored in the pvals component of the prediction component of the value produced by predict.asreml. This data.frame is added to the alldiffs object as a list component called backtransforms.

The printing of the components produced is controlled by the tables argument.

Usage

```
predictparallel.asreml(classify, term = NULL, asreml.obj = NULL, titles = NULL,
                       x.num = NULL, x.fac = NULL,
                       x.pred.values = NULL, x.plot.values = NULL,
                       error.intervals = "Confidence", avsed.tolerance = 0.25,
                       pairwise = TRUE, tables = "all" , levels.length = NA,
                       transform.power = 1, offset = 0, scale = 1,
                       inestimable.rm = TRUE, wald.tab = NULL,
                       alpha = 0.05, dDF.na = "residual",  dDF.values = NULL,
                       trace = FALSE, ...)
```

Arguments

classify	a character string giving the variables that define the margins of the multiway table to be predicted. Multiway tables are specified by forming an interaction type term from the classifying variables, that is, separating the variable names with the : operator.
term	a character string giving the variables that define the term that was fitted using asreml and that corresponds to classify. It only needs to be specified when it is different to classify.
asreml.obj	asreml object for a fitted model.
titles	A list, each component of which is named for an object name and contains a character string giving a title to use in output (e.g. tables and graphs) for the object. Here they will be used for table headings.
x.num	A character string giving the name of the numeric covariate that (i) corresponds to x.fac, (ii)is potentially included in terms in the fitted model, and (iii) which corresponds to the x-axis variable. It should have the same number of unique values as the number of levels in x.fac.
x.fac	A character string giving the name of the factor that (i) corresponds to x.num, (ii) is potentially included in terms in the fitted model, and (iii) which corresponds to the x-axis variable. It should have the same number of levels as the number of unique values in x.num. The levels of x.fac must be in the order in which they are to be plotted - if they are dates, then they should be in the form yyyymmdd which can be achieved using as.Date.

`x.pred.values` The values of `x.num` for which predicted values are required.

`x.plot.values` The actual values to be plotted on the x axis. They are needed when values different to those in `x.num` are to be plotted or `x.fac` is to be plotted because there is no `x.num` term corresponding to the same term with `x.fac`.

`error.intervals`
A character string indicating the type of error interval, if any, to calculate and plot in order to indicate uncertainty in the results. Possible values are `"none"`, `"StandardError"`, `"Confidence"` and `"halfLeastSignificant"`. The default is for confidence limits to be used. The `"halfLeastSignificant"` option results in half the mean Least Significant Difference (LSD) being added and subtracted to the predictions, the mean LSD being calculated using the average of the standard errors of all pairwise differences (SEDs) between the predictions. However, if the range of the SEDs divided by the average of the SEDs exceeds `avsed.tolerance`, calculations and plotting will revert to confidence intervals. Also, half LSDs cannot be used for backtransformed values and so confidence intervals will be used instead.

`avsed.tolerance`
The values of the range of the SEDs divided by the average of the SEDs that, if exceeded, will cause calculations and plotting to revert to confidence intervals. It should be a value between 0 and 1.

`pairwise` A logical indicating whether all pairwise differences of the `predictions` and their standard errors and p-values are to be computed and stored. If `tables` is equal to `"differences"` or `"all"` or `error.intervals` is equal to `"halfLeastSignificant"`, they will be stored irrespective of the value of `pairwise`.

`tables` A `character vector` containing a combination of `none`, `predictions`, `backtransforms`, `differences`, `p.differences`, `sed`, `LSD` and `all`. These nominate which components of the `alldiffs` object to print.

`levels.length` The maximum number of characters from the the levels of factors to use in the row and column labels of the tables of pairwise differences and their p-values and standard errors.

`transform.power`
A number specifying the power of a transformation, if one has been applied to the response variable. Unless it is equal to 1, the default, back-transforms of the predictions will be obtained and presented in tables or graphs as appropriate. The back-transformation raises the predictions to the power equal to the reciprocal of `transform.power`, unless it equals 0 in which case the exponential of the predictions is taken.

`offset` A number that has been added to each value of the response after any scaling and before applying any power transformation.

`scale` A number by which each value of the response has been multiply before adding any offset and applying any power transformation.

`inestimable.rm` A `logical` indicating whether rows for predictions that are not estimable are to be removed from the components of the`alldiffs` object.

`wald.tab` a `data.frame` containing the pseudo-anova table for the fixed terms produced by a call to `wald.asreml`. The main use of it here is in determinining the degrees of freedom of the standard errors of the predictions. denominator degrees of freedom when p-values or confidence intervals are to be calculated.

alpha	the significance level for a test or one minus the confidence level for confidence intervals.
dDF.na	the method to use to obtain approximate denominator degrees of freedom. when the numeric or algebraic methods produce an NA. Consistent with when no denDF are available, the default is "residual" and so the residual degrees of freedom from asreml.obj$nedf are used. If dDF.na = "none", no subtitute denominator degrees of freedom are employed; if dDF.na = "maximum", the maximum of those denDF that are available, excluding that for the Intercept, is used; if all denDF are NA, asreml.obj$nedf is used. If dDF.na = "supplied", a vector of values for the denominator degrees of freedom is to be supplied in dDF.values. Any other setting is ignored and a warning message produced. Generally, substituting these degrees of freedom is anticonservative in that it is likely that the degrees of freedom used will be too large.
dDF.values	A vector of values to be used when dDF.na = "supplied". Its values will be used when denDF in a test for a fixed effect is NA. This vector must be the same length as the number of fixed terms, including (Intercept) whose value could be NA.
trace	if TRUE then partial iteration details are displayed when ASReml-R functions are invoked; if FALSE then no output is displayed.
...	further arguments passed to predict.asreml.

Value

An alldiffs object with predictions and their standard errors and, depending on the settings of the arguments, all pairwise differences between predictions, their standard errors and p-values and LSD statistics. If power.transform is not one, it will contain a data.frame with the backtransformed predictions. If error.intervals is not "none", then the predictions component and, if present, the backtransforms component will contain columns for the lower and upper values of the limits for the interval. The names of these columns will consist of three parts separated by full stops: 1) the first part will be lower or upper; 2) the second part will be one of Confidence, StandardError or halfLeastSignificant; 3) the third component will be limits.

The name of the response, the term, the classify and tdf, as well as the degrees of freedom of the standard error, will be set as attributes to the object.

See Also

alldiffs, print.alldiffs, predictiondiffs.asreml, pred.present.asreml, as.Date, predictionplot.asreml, pred.present.asreml, predict.asreml

Examples

```
## Not run:
data(WaterRunoff.dat)
current.asr <- asreml(fixed = pH ~ Benches + (Sources * (Type + Species)),
                      random = ~ Benches:MainPlots,
                      keep.order=TRUE, data= WaterRunoff.dat)
current.asrt <- asrtests(current.asr, NULL, NULL)
diffs <- predictparallel.asreml(classify = "Sources:Type",
```

```
                                asreml.obj = current.asr,
                                x.num = "xDay", x.fac = "Date",
                                x.pred.values=sort(unique(WaterRunoff.dat$xDay)),
                                x.plot.values=c(0,28,56,84),
                                wald.tab = current.asrt$wald.tab,
                                present = c("Sources", "Type", "Species"))

## End(Not run)
```

`print.alldiffs` *Prints the values in an* `alldiffs` *object in a nice format.*

Description

Prints the predictions and standard errors from using asreml to fit models in the same way as `asreml` prints them. Also prints out all pairwise differences between the predictions to 2 signigicant figures, along with their p-values and standard errors to 4 decimal places. If LSDs are requested the mean. minimum and maximum LSDs will be printed.

Usage

```
## S3 method for class 'alldiffs'
print(x, which = "all", ...)
```

Arguments

`x`	A `list` returned from `asreml.predictiondiffs`.
`which`	A `character vector` containing a combination of `predictions`, `backtransforms`, `differences`, `p.differences`, `sed`, `LSD` and `all`. These nominate which components of the `alldiffs` object to print.
`...`	further arguments passed to or from other methods.

Value

No value is returned, but the elements of the list in `x`, returned from `asreml.predictiondiffs`, are printed.

See Also

`alldiffs`, `predictiondiffs.asreml`

Examples

```
## Not run:
print.alldiffs(predictiondiffs.asreml(asreml=splitplot.asr, wald=wald,
               present=c("Type","Sources","Species"),
               factors=list("Species")))

## End(Not run)
```

print.asrtests *Prints the values in an* asrtests *object*

Description

Prints a summary of the asreml object and the test.summary data.frame that are stored in the asrtests object.

Usage

```
## S3 method for class 'asrtests'
print(x, which = "all", ...)
```

Arguments

x	An asrtests object.
which	Which elements of the asrtests object to print. Possible values are some combination of asremlsummary, pseudoanova, testsummary and all.
...	further arguments passed to print.

Value

No value is returned, but the elements of the list in x are printed.

See Also

asrtests, asremlPlus-package

Examples

```
## Not run:
data(Wheat.dat)
current.asr <- asreml(yield ~ Rep + WithinColPairs + Variety,
                      random = ~ Row + Column + units,
                      rcov = ~ ar1(Row):ar1(Column),
                      data=Wheat.dat)
current.asrt <- asrtests(current.asr, NULL, NULL)
current.asrt <- rmboundary.asrtests(current.asrt)
# Test Row autocorrelation
current.asrt <- testrcov.asrtests("~ Row:ar1(Column)", current.asrt,
                                  label="Row autocorrelation", simpler=TRUE)
print(current.asrt)

## End(Not run)
```

recalc.wald.tab.asrtests *Recalculates the denDF, F.inc and P values for a table of Wald test statistics obtained using* wald.asreml

Description

If some or all denDF are not available, either because they are NA or because F.inc values were not calculated, this function allows the user to specify how approximate denDF values are to be obtained. This is done through the dDF.na and dDF.values arguments. Note that if denDF values are available in the Wald table then only those that are NA will be replaced.

Usage

```
recalc.wald.tab.asrtests(asrtests.obj, recalc.wald = FALSE,
                         denDF="default", dDF.na = "none",
                         dDF.values = NULL, trace = FALSE, ...)
```

Arguments

asrtests.obj	an asrtests object for a fitted model that is a list containing an asreml object, a wald.tab data.frame with 4 columns, and a data.frame with 5 columns that records any previous changes and tests in the fitted model.
recalc.wald	a logical indicating whether to call wald.asreml to recalculate the pseudoanova table for the model fit stored in the asreml object contained in asrtests.
denDF	Specifies the method to use in computing approximate denominator degrees of freedom when wald.asreml is called. Can be none to suppress the computations, numeric for numerical methods, algebraic for algebraic methods or default, the default, to autommatically choose numeric or algebraic computations depending on problem size. The denominator degrees of freedom are calculated according to Kenward and Roger (1997) for fixed terms in the dense part of the model.
dDF.na	The method to use to obtain substitute denominator degrees of freedom. when the numeric or algebraic methods produce an NA. If dDF.na = "none", no substitute denominator degrees of freedom are employed; if dDF.na = "residual", the residual degrees of freedom from asreml.obj$nedf are used; if dDF.na = "maximum", the maximum of those denDF that are available, excluding that for the Intercept, is used; if all denDF are NA, asreml.obj$nedf is used. If dDF.na = "supplied", a vector of values for the denominator degrees of freedom is to be supplied in dDF.values. Any other setting is ignored and a warning message produced. Generally, substituting these degrees of freedom is anticonservative in that it is likely that the degrees of freedom used will be too large.
dDF.values	A vector of values to be used when dDF.na = "supplied". Its values will be used when denDF in a test for a fixed effect is NA. This vector must be the same length as the number of fixed terms, including (Intercept) whose value could be NA.

trace	if TRUE then partial iteration details are displayed when ASReml-R functions are invoked; if FALSE then no output is displayed.
...	further arguments passed to asreml and to wald.asreml.

Value

A wald.tab: a 4-column data.frame containing a pseudo-anova table for the fixed terms produced by wald.asreml.

See Also

asrtests, testranfix.asrtests

Examples

```
## Not run:
  wald.tab <- recalc.wald.tab.asrtests(current.asrt,
                                       dDF.na = "supplied",
                                       dDF.values = c(NA,rep(c(330,346), c(4,3))))

## End(Not run)
```

---

reml.lrt.asreml *Performs REML likelihood ratio test.*

Description

Extracts the REML log likelihood and number of variance parameters from two asreml objects. It assumes that the second asreml object is the result of fitting a model that is a reduced version of the model for the first object. In the case where the reduced model is obtained by setting positively-constrained variance parameters in the full model to zero, the positive.zero argument should be set to TRUE so that the p-value is computed using a mixture of chi-square distributions as described in Self and Liang (1987).

The function shecks that the models do not differ in either their fixed or sparse models.

The function reml.lrt is provided for backwards compatibility.

Usage

```
reml.lrt.asreml(full.asreml.obj, reduced.asreml.obj,
                positive.zero = FALSE, bound.test.parameters = "none")
```

Arguments

`full.asreml.obj`
: `asreml` object for the full model.

`reduced.asreml.obj`
: `asreml` object for the reduced model.

`positive.zero`
: Indicates whether the hypothesized values for the variance components being tested are on the boundary of the parameter space. For example, this is true for positively-constrained variance components that, under the reduced model, are zero. This argument does not need to be set if `bound.test.parameters` is set.

`bound.test.parameters`
: Indicates whether for the variance components being tested, at least some of the hypothesized values are on the boundary of the parameter space. The default is `"none"`. Other possibilities are `"onlybound"` and `"one-and-one"`. The latter signifies that there are two parameters being tested, one of which is bound and the other is not. For example, the latter is true for testing a covariance and a positively-constrained variance component that, under the reduced model, are zero.

Value

A data frame containing the log of the likelihood ratio, its degrees of freedom and its p-value.

Note

The degrees of freedom for the test are computed as the difference between the two models in the number of variance parameters that are unfixed, nonsingular and unconstrained by relationships among them.

This procedure is only approriate when the null hypothesis is that all parameters are on the boundary of the parameter space or that all parameters are in the interior of the parameter space. Mixed cases have been discussed by Self and Liang (1987), but are not implemented here.

References

Self, S.G., and Liang, K-Y. (1987) Asymptotic Properties of Maximum Likelihood Estimators and Likelihood Ratio Tests Under Nonstandard Conditions. *Journal of the American Statistical Association*, **82**, 605-10.

See Also

`info.crit.asreml`, `testranfix.asrtests`

Examples

```
## Not run:
    reml.lrt.asreml(ICV.max, ICV.red, bound.test.parameters = "onlybound")

## End(Not run)
```

`rmboundary.asrtests` *Removes any boundary or singular variance components from the fit stored in* `asreml.obj` *and records their removal in a data.frame.*

Description

Any terms specified in the random model that are estimated on the boundary or are singular and can be removed are removed from the fit stored in an `asreml` object. Terms that specify multiple parameters in the random model cannot be removed (e.g. terms specified using the `at` function with more than one level of the factor) and terms in `rcov` model are not removed. Terms that can be removed are selected for removal in the following order based on whether they involve: (i) a `dev` function, (ii) only factors, (iii) an `spl` function, (iv) a `pol` function and (v) a `lin` function or a variable that is an `integer` or a `numeric`. It should be noted that this order of removal presumes that random deviation terms are specified via the `dev` function rather than via a random factor. Once the earliest of thes above classes with a boundary term is identified, a term within this class is selected for removal. For all classes, except for factor-only terms, the smallest term with the largest number of variables/factors is removed. Amongst factor-only terms, the smallest term with the smallest number of variables/factors is removed. After each variance component is removed, a row for it is added to the `test.summary` `data.frame` and the model refitted. If there are further boundary or singular terms, one is removed using the above strategy. This process continues until there are no further boundary or singular variance components that are removable. Other types of boundary or singular terms, which cannot be removed, are reported in warning messages.

Usage

```
rmboundary.asrtests(asrtests.obj, trace = FALSE, update = TRUE,
                    set.terms = NULL, ignore.suffices = TRUE,
                    constraints = "P", initial.values = NA, ...)
```

Arguments

`asrtests.obj`	an `asrtests` object for a fitted model that is a list containing an `asreml` object, a `wald.tab` `data.frame` with 4 columns, and a `data.frame` with 5 columns that records any previous changes and tests in the fitted model.
`trace`	if TRUE then partial iteration details are displayed when ASReml-R functions are invoked; if FALSE then no output is displayed.
`update`	if `TRUE` then `update.asreml` is called to fit the model with any boundary terms removed. In doing this the arguments `R.param` and `G.param` are set to those in the `asreml` object stored in `asrtests.obj` so that the values from the previous model are used as starting values. If `FALSE` then a call is made to `asreml` in which the only changes from the previous call are that (i) the terms for boundary variance components are removed from the models and (ii) modifications specified via `...` are made.
`set.terms`	a character vector specifying the terms that are to have constraints and/or initial values set prior to fitting.

`ignore.suffices`
: a logical vector specifying whether the suffices of the `asreml`-assigned names of the variance terms (i.e. the information to the right of an "!", other than "R!") is to be ignored in matching elements of `terms`. If `TRUE` for an element of `terms`, the suffices are stripped from the `asreml`-assigned names. If `FALSE` for an element of `terms`, the element must exactly match an `asreml`-assigned name for a variance term. This vector must be of length one or the same length as `terms`. If it is of length one then the same action is applied to the `asreml`-assigned suffices for all the terms in `terms`.

`constraints`
: a character vector specifying the constraints to be applied to the terms specified in `terms`. This vector must be of length one or the same length as `terms`. If it is of length one then the same constraint is applied to all the terms in `terms`. If any of the constraints are equal to NA then they are left unchanged for those terms.

`initial.values`
: a character vector specifying the initial values for the terms specified in `terms`. This vector must be of length one or the same length as `terms`. If it is of length one then the same initial value is applied to all the terms in `terms`. If any of the initial.values are equal to NA then they are left unchanged for those terms.

`...`
: further arguments passed to `asreml`.

Value

An `asrtests` object, which is a list containing:

1. `asreml.obj`: an `asreml` object containing the fit of the model after all boundary and singular terms have been removed;
2. `wald.tab`: a 4-column `data.frame` containing a pseudo-anova table for the fixed terms produced by `wald.asreml`;
3. `test.summary`: a `data.frame` with columns `term`, `DF`, `denDF`, `p` and `action`. A row is added to it for each of the boundary terms removed, the row containing the name of the term, one for the DF, NA for the p-value and `Boundary` for the action.

See Also

`asrtests`, `addrm.terms.asrtests`, `testranfix.asrtests`, `testrcov.asrtests`, `newfit.asreml`, `sig.devn.reparam.asrtests`, `choose.model.asrtests`

Examples

```
## Not run:
current.asrt <- rmboundary.asrtests(current.asrt)

## End(Not run)
```

setvarianceterms.asreml *allows the seting of constraints and initial values for terms in the* random *and* rcov *arguments of an* asreml *call, with the resulting call being evaluated.*

Description

Takes an unevaluated call and evaluates the call after setting the constraints and initial values for the terms specified in terms. The elements of terms are matched with those generated by asreml and used, for example, in the varcomp component of a summary.asreml object. These names generally include descriptive suffices. To match an element of terms that includes such a suffix, set ignore.suffices to FALSE so that a literal match between the element and the assigned names is sought.

Usage

```
setvarianceterms.asreml(call, terms, ignore.suffices = TRUE,
                        constraints = "P", initial.values = NA, ...)
```

Arguments

call	an unevaluated call to asreml. One way to create such a call is to use the call function with its name argument set to "asreml". Another is to obtain it from the call component of an asreml object (e.g. call <- asreml.obj$call).
terms	a character vector specifying the terms that are to have constraints and/or initial values specified.
ignore.suffices	a logical vector specifying whether the suffices of the asreml-assigned names of the variance terms (i.e. the information to the right of an "!", other than "R!") is to be ignored in matching elements of terms. If TRUE for an element of terms, the suffices are stripped from the asreml-assigned names. If FALSE for an element of terms, the element must exactly match an asreml-assigned name for a variance term. This vector must be of length one or the same length as terms. If it is of length one then the same action is applied to the asreml-assigned suffices for all the terms in terms.
constraints	a character vector specifying the constraints to be applied to the terms specified in terms. This vector must be of length one or the same length as terms. If it is of length one then the same constraint is applied to all the terms in terms. If any of the constraints are equal to NA then they are left unchanged for those terms.
initial.values	a character vector specifying the initial values for the terms specified in terms. This vector must be of length one or the same length as terms. If it is of length one then the same initial value is applied to all the terms in terms. If any of the initial.values are equal to NA then they are left unchanged for those terms.
...	additional arguments to be added to the call, or arguments in the call with changed values.

Value

An asreml object.

References

Butler, D. G., et al. (2010). *Analysis of Mixed Models for S language environments: ASReml-R reference manual.* Brisbane, DPI Publications.

See Also

update.asreml

Examples

```
## Not run:
   m1.call <- call("asreml",
                   fixed = Height ~ (Block + Irrig)*csDay.num,
                   random= ~ spl(csDay.num)/(Irrig+Block)
                             + dev(csDay.num)
                             + str(~Block:Plot/csDay.num, ~us(2):id(20))
                             + Block:Plot:spl(csDay.num),
                   data=quote(dat)) ##use quote to stop evaluation of dat here
   terms <- c("Block:Plot+Block:Plot:csDay.num!us(2).2:1", "R!variance")
   m1.asreml <- setvarianceterms.asreml(m1.call, terms,
                                        constraints=c("U","P"),
                                        initial=c(NA,3),
                                        ignore.suffices=c(FALSE,TRUE))
   summary(m1.asreml)

## End(Not run)
```

---

sig.devn.reparam.asrtests

This function reparamterizes each random (deviations) term involving devn.fac *to a fixed term and ensures that the same term, with* trend.num *replacing* devn.fac*, is included if any other term with* trend.num *is included in* terms*.*

Description

This function reparamterizes each random (deviations) term involving devn.fac to a fixed term and ensures that the same term with trend.num replacing devn.fac is included if any other term with trend.num is included in terms. It also ansures that any term with spl{trend.num} replacing devn.fac in a term being reparameterized is removed from the model.

Usage

```
sig.devn.reparam.asrtests(terms = NULL, asrtests.obj,
                          trend.num = NULL, devn.fac = NULL,
                          denDF = "default", trace = FALSE, update = TRUE,
                          set.terms = NULL, ignore.suffices = TRUE,
                          constraints = "P", initial.values = NA,...)
```

Arguments

terms	a character string vector giving the terms that are to be reparameterized.
asrtests.obj	an asrtests object for a fitted model that is a list containing an asreml object, a wald.tab data.frame with 4 columns, and a data.frame with 5 columns that records any previous changes and tests in the fitted model.
trend.num	A character string giving the name of the numeric covariate that corresponds to devn.fac and is potentially included in terms in the fitted model.
devn.fac	A character string giving the name of the factor that corresponds to trend.num and is included in terms in the fitted model.
denDF	Specifies the enthod to use in computing approximate denominator degrees of freedom when wald.asreml is called. Can be none to suppress the computations, numeric for numerical methods, algebraic for algebraic methods or default, the default, to autommatically choose numeric or algebraic computations depending on problem size. The denominator degrees of freedom are calculated according to Kenward and Roger (1997) for fixed terms in the dense part of the model.
trace	if TRUE then partial iteration details are displayed when ASReml-R functions are invoked; if FALSE then no output is displayed.
update	if TRUE then update.asreml is called in removing and adding terms to the model. In doing this the arguments R.param and G.param are set to those in the asreml object stored in the supplied asrtests.obj so that the values from the previous model are used as starting values. If FALSE then calls are made to asreml in which the only changes from the previous call are (i) that the models are updated and (ii) modifications specified via ... are made.
set.terms	a character vector specifying the terms that are to have constraints and/or initial values set prior to fitting.
ignore.suffices	a logical vector specifying whether the suffices of the asreml-assigned names of the variance terms (i.e. the information to the right of an "!", other than "R!") is to be ignored in matching elements of terms. If TRUE for an element of terms, the suffices are stripped from the asreml-assigned names. If FALSE for an element of terms, the element must exactly match an asreml-assigned name for a variance term. This vector must be of length one or the same length as terms. If it is of length one then the same action is applied to the asreml-assigned suffices for all the terms in terms.
constraints	a character vector specifying the constraints to be applied to the terms specified in terms. This vector must be of length one or the same length as terms. If it is

of length one then the same constraint is applied to all the terms in terms. If any of the constraints are equal to NA then they are left unchanged for those terms.

initial.values a character vector specifying the initial values for the terms specified in terms. This vector must be of length one or the same length as terms. If it is of length one then the same initial value is applied to all the terms in terms. If any of the initial.values are equal to NA then they are left unchanged for those terms.

... further arguments passed to asreml via addrm.terms.asrtests.

Value

An asrtests object, which is a list containing:

1. asreml.obj: an asreml object containing the fit of the model after all boundary and singular terms have been removed;
2. wald.tab: a 4-column data.frame containing a pseudo-anova table for the fixed terms produced by wald.asreml;
3. test.summary: a data.frame with columns term, DF, denDF, p and action. It contains a row for each term that is dropped, added or tested or a note that several terms have been added or removed.

See Also

asrtests, addrm.terms.asrtests, testranfix.asrtests, testrcov.asrtests, newfit.asreml, choose.model.asrtests

Examples

```
## Not run:
  data(WaterRunoff.dat)
  current.asr <- asreml(fixed = log.Turbidity ~ Benches + Sources + Type + Species +
                          Sources:Type + Sources:Species + Sources:Species:xDay +
                          Sources:Species:Date,
                        data = WaterRunoff.dat, keep.order = TRUE)
  current.asrt <- asrtests(current.asr, NULL, NULL)

  #Examine terms that describe just the interactions of Date and the treatment factors
  terms.treat <- c("Sources", "Type", "Species", "Sources:Type", "Sources:Species")
  date.terms <- sapply(terms.treat,
                       FUN=function(term){paste("Date:",term,sep="")},
                       simplify=TRUE)
  date.terms <- c("Date", date.terms)
  date.terms <- unname(date.terms)
  treat.marginality <-  matrix(c(1,0,0,0,0,0, 1,1,0,0,0,0,  1,0,1,0,0,0,
                                 1,0,1,1,0,0, 1,1,1,0,1,0, 1,1,1,1,1,1), nrow=6)
  rownames(treat.marginality) <- date.terms
  colnames(treat.marginality) <- date.terms
  choose <- choose.model.asrtests(treat.marginality, current.asrt, denDF="algebraic")
  current.asrt <- choose$asrtests.obj
  current.asr <- current.asrt$asreml.obj
  sig.date.terms <- choose$sig.terms
```

```
  #Remove all Date terms left in the fixed model
  terms <- "(Date/(Sources * (Type + Species)))"
  current.asrt <- addrm.terms.asrtests(terms, current.asrt)
  #if there are significant date terms, reparameterize to xDays + spl(xDays) + Date
  if (length(sig.date.terms) != 0)
  { #add lin + spl + devn for each to fixed and random models
    trend.date.terms <- sapply(sig.date.terms,
                               FUN=function(term){sub("Date","xDay",term)},
                               simplify=TRUE)
    trend.date.terms <- paste(trend.date.terms,  collapse=" + ")
    current.asrt <- addrm.terms.asrtests(trend.date.terms, current.asrt, add=TRUE)
    trend.date.terms <- sapply(sig.date.terms,
                               FUN=function(term){sub("Date","spl(xDay)",term)},
                               simplify=TRUE)
    trend.date.terms <- c(trend.date.terms, sig.date.terms)
    trend.date.terms <- paste(trend.date.terms,  collapse=" + ")
    current.asrt <- addrm.terms.asrtests(trend.date.terms, current.asrt,
                                         add=TRUE, random=TRUE)
    current.asrt <- rmboundary.asrtests(current.asrt)
  }

  #Now test terms for sig date terms
  spl.terms <- sapply(terms.treat,
                      FUN=function(term){paste("spl(xDay):",term,sep="")},
                      simplify=TRUE)
  spl.terms <- c("spl(xDay)",spl.terms)
  lin.terms <- sapply(terms.treat,
                      FUN=function(term){paste(term,":xDay",sep="")},
                      simplify=TRUE)
  lin.terms <- c("xDay",lin.terms)
  systematic.terms <- c(terms.treat, lin.terms, spl.terms, date.terms)
  systematic.terms <- unname(systematic.terms)
  treat.marginality <-  matrix(c(1,0,0,0,0,0, 1,1,0,0,0,0,  1,0,1,0,0,0,
                                 1,0,1,1,0,0, 1,1,1,1,1,0, 1,1,1,1,1,1), nrow=6)
  systematic.marginality <- kronecker(matrix(c(1,0,0,0, 1,1,0,0,
                                               1,1,1,0, 1,1,1,1), nrow=4),
                                      treat.marginality)
  systematic.marginality <- systematic.marginality[-1, -1]
  rownames(systematic.marginality) <- systematic.terms
  colnames(systematic.marginality) <- systematic.terms
  choose <- choose.model.asrtests(systematic.marginality, current.asrt,
                                  denDF="algebraic", pos=TRUE)
  current.asrt <- choose$asrtests.obj

  #Check if any deviations are significant and, for those that are, go back to
  #fixed dates
  current.asrt <- sig.devn.reparam.asrtests(choose$sig.terms, current.asrt,
                                            trend.num = "xDay", devn.fac = "Date",
                                            denDF = "algebraic")

## End(Not run)
```

`simulate.asreml` *Produce sets of simulated data from a multivariate normal distribtion and save quantites related to the simulated data*

Description

Produce a set of simulated data corresponding to an `asreml` model, along with its fitted values and residuals. A variance matrix `V`, corresponding to the `random` and `rcov` models must be supplied. What to save is specified by the `which` argument.

Usage

```
## S3 method for class 'asreml'
simulate(object, nsim=100, seed = NULL, means=NULL, V, tolerance = 1E-10,
         update = TRUE, trace = FALSE, which="data", units = "ignore", ...)
```

Arguments

`object`	An `asreml` object from a call to `asreml` in which the `data` argument has been set.
`means`	The `vector` of means to be used in generating simulated data sets. If it is `NULL`, the fitted values based on `object` are used. It must be the same length as the response variable for `object`.
`V`	The fitted variance `matrix`, i.e. having the pattern and values that conform to the model fit stored in the supplied `object`.
`nsim`	The number of data sets to be simulated.
`seed`	A single value, interpreted as an integer, that specifies the starting value of the random number generator.
`tolerance`	The value such that eigenvalues less than it are consdered to be zero.
`update`	if `TRUE` then the arguments `R.param` and `G.param` are set to those in the `asreml` object supplied in `object` so that the values from the original model are used as starting values. If `FALSE` then calls are made to `asreml` in which the only changes from the previous call are (i) the model is fitted to simulated data and (ii) modifications specified via `...` are made, except that changes cannot be made to any of the models.
`trace`	if `TRUE` then partial iteration details are displayed when ASReml-R functions are invoked; if `FALSE` then no output is displayed.
`which`	The quantites from the simulated data set to be stored. Any combination of `"response"`, `"residuals"` and `"fitted"`, or `"all"`. If `residuals` and/or `fitted` is specified, those for the analysis stored in `object` will be added to the `data.frame` nominated in the `data` argument of `object` and the modified `data.frame` added as a component named `data` in the `list` that is the `value` returned by the `function`.

units	A character indicating whether the BLUPs for units are added to the residuals when this reserved factor is included in the random model. Possible values are addtoresiduals and ignore.
...	Other arguments that are passed down to the function asreml. Changes to the models are not allowed. Other changes are dangerous and generally should be avoided.

Details

Generate nsim set of data and analyse them using asreml using the model in object. Note, if the analysis for a data set does not converge in maxiter iterations, it is discarded and a replacement data set generated. The value of maxiter can be specified in the call to simulate.asreml. The fitted values and residuals are extracted as required. If aom = TRUE when the simulated data are analysed, standardised conditional residuals are stored. If which includes residuals or fitted, the specified quantities for the observed data are added to the data.frame on which the fit in object is based.

Value

A list with the following components whose presence depends on the setting of which:

1. **observed:** present if which includes residuals or fitted, in which case it will be the data.frame on which the fit in object is based, with residuals and/or fitted.
2. **data:** present if which includes data, a data.frame containing the simulated data sets.
3. **fitted:** present if which includes fitted, a data.frame containing the fitted values from the analyses of the simulated data sets.
4. **residuals:** present if which includes residuals, a data.frame containing the residuals from the analyses of the simulated data sets.

Author(s)

Chris Brien

See Also

asreml, variofaces.asreml, plotvariofaces.asreml.

Examples

```
## Not run:
data(Wheat.dat)
current.asr <- asreml(yield ~ Rep + WithinColPairs + Variety,
                      random = ~ Row + Column + units,
                      rcov = ~ ar1(Row):ar1(Column),
                      data=Wheat.dat)
current.asrt <- asrtests(current.asr, NULL, NULL)
current.asrt <- rmboundary.asrtests(current.asrt)
# Form variance matrix based on estimated variance parameters
s2 <- current.asr$sigma2
gamma.Row <- current.asr$gammas[1]
gamma.unit <- current.asr$gammas[2]
```

```
rho.r <- current.asr$gammas[4]
rho.c <- current.asr$gammas[5]
row.ar1 <- mat.ar1(order=10, rho=rho.r)
col.ar1 <- mat.ar1(order=15, rho=rho.c)
V <- gamma.Row * fac.sumop(Wheat.dat$Row) +
     gamma.unit * diag(1, nrow=150, ncol=150) +
     mat.dirprod(col.ar1, row.ar1)
V <- s2*V

#Produce residuals from 100 simulated data sets
resid <- simulate(current.asr, V=V, which="residuals")

## End(Not run)
```

---

testranfix.asrtests *Tests for a single fixed or random term in model fitted using* asreml

---

Description

Tests for a single term, using a REML LRT for a random term or based on Wald statistics for a fixed term. The term must be in the fitted model. A random term is removed from the model fit and a REML likelihood ratio test is performed using reml.lrt.asreml. It compares the fit of the model in asreml.obj and the newly fitted model without the term. If the newly fitted model is retained, any boundary terms are then removed using rmboundary.asrtests. For a fixed term, the probability of the Wald statistics is extracted from the pseudo-anova table produced by wald.asreml. If this is available in the asrtests object, it is used; otherwise wald.asreml is called to add it to the asrtests object. Whether nonsignificant terms are dropped is controlled by drop.ran.ns for random terms and drop.fix.ns for fixed terms. A row is added to the test.summary data.frame for the term that is tested.

Usage

```
testranfix.asrtests(term=NULL, asrtests.obj, alpha = 0.05,
                    drop.ran.ns = TRUE,
                    positive.zero = FALSE, bound.test.parameters = "none",
                    drop.fix.ns = FALSE, denDF="default", dDF.na = "none",
                    dDF.values = NULL, trace = FALSE, update = TRUE,
                    set.terms = NULL, ignore.suffices = TRUE,
                    constraints = "P", initial.values = NA, ...)
```

Arguments

term	a single model term that is valid in asreml, stored as a character.
asrtests.obj	an asrtests object for a fitted model that is a list containing an asreml object, a wald.tab data.frame with 4 columns, and a data.frame with 5 columns that records any previous changes and tests in the fitted model.
alpha	the significance level for the test.

drop.ran.ns a logical indicating whether to drop a random term from the model when it is nonsignificant .

positive.zero Indicates whether the hypothesized values for the variance components being tested are on the boundary of the parameter space. For example, this is true for positively-constrained variance components that, under the reduced model, are zero. This argument does not need to be set if bound.test.parameters is set.

bound.test.parameters
Indicates whether for the variance components being tested, at least some of the hypothesized values are on the boundary of the parameter space. The default is "none". Other possibilities are "onlybound" and "one-and-one". The latter signifies that there are two parameters being tested, one of which is bound and the other is not. For example, the latter is true for testing a covariance and a positively-constrained variance component that, under the reduced model, are zero.

drop.fix.ns a logical indicating whether to drop a fixed term from the model when it is nonsignificant

denDF Specifies the enthod to use in computing approximate denominator degrees of freedom when wald.asreml is called. Can be none to suppress the computations, numeric for numerical methods, algebraic for algebraic methods or default, the default, to autommatically choose numeric or algebraic computations depending on problem size. The denominator degrees of freedom are calculated according to Kenward and Roger (1997) for fixed terms in the dense part of the model.

dDF.na the method to use to obtain substitute denominator degrees of freedom. when the numeric or algebraic methods produce an NA. If dDF.na = "none", no subtitute denominator degrees of freedom are employed; if dDF.na = "residual", the residual degrees of freedom from asreml.obj$nedf are used; if dDF.na = "maximum", the maximum of those denDF that are available, excluding that for the Intercept, is used; if all denDF are NA, asreml.obj$nedf is used. If dDF.na = "supplied", a vector of values for the denominator degrees of freedom is to be supplied in dDF.values. Any other setting is ignored and a warning message produced. Generally, substituting these degrees of freedom is anticonservative in that it is likely that the degrees of freedom used will be too large.

dDF.values A vector of values to be used when dDF.na = "supplied". Its values will be used when denDF in a test for a fixed effect is NA. This vector must be the same length as the number of fixed terms, including (Intercept) whose value could be NA.

trace if TRUE then partial iteration details are displayed when ASReml-R functions are invoked; if FALSE then no output is displayed.

update if TRUE then update.asreml is called to fit the model to be tested. In doing this the arguments R.param and G.param are set to those in the asreml object stored in asrtests.obj so that the values from the previous model are used as starting values. If FALSE then a call is made to asreml in which the only changes from the previous call are that (i) models are modifed for the supplied terms and (ii) modifications specified via ... are made.

set.terms a character vector specifying the terms that are to have constraints and/or initial values set prior to fitting.

ignore.suffices
: a logical vector specifying whether the suffices of the asreml-assigned names of the variance terms (i.e. the information to the right of an "!", other than "R!") is to be ignored in matching elements of terms. If TRUE for an element of terms, the suffices are stripped from the asreml-assigned names. If FALSE for an element of terms, the element must exactly match an asreml-assigned name for a variance term. This vector must be of length one or the same length as terms. If it is of length one then the same action is applied to the asreml-assigned suffices for all the terms in terms.

constraints
: a character vector specifying the constraints to be applied to the terms specified in terms. This vector must be of length one or the same length as terms. If it is of length one then the same constraint is applied to all the terms in terms. If any of the constraints are equal to NA then they are left unchanged for those terms.

initial.values
: a character vector specifying the initial values for the terms specified in terms. This vector must be of length one or the same length as terms. If it is of length one then the same initial value is applied to all the terms in terms. If any of the initial.values are equal to NA then they are left unchanged for those terms.

...
: further arguments passed to asreml and to wald.asreml.

Value

An asrtests object, which is a list containing:

1. asreml.obj: an asreml object containing the fit after the term has been tested; it will be a new model if the term is nonsignificant and the appropriate argument out of drop.ran.ns and drop.fix.ns is TRUE;
2. wald.tab: a 4-column data.frame containing a pseudo-anova table for the fixed terms produced by wald.asreml;
3. test.summary: a data.frame with columns term, DF, denDF, p and action. A row is added to it for each term that is tested, the row containing the name of the term, the degrees of freedom (numerator DF for a Wald test and the number of extra paramters for a REML ratio tests), the p-value and a for the action taken. Possible codes are: Dropped, Retained, Significant, Nonsignificant, Absent, Added, Removed and Boundary. If the changed model did not converge, Unconverged will be added to the code. Note that the logical asreml.obj$converge also reflects whether there is convergence.

If the term is not in the model, then the supplied asreml object will be returned. Also, reml.test will have the likelihood ratio and the p-value set to NA and the degrees of freedom to zero. Similarly, the row of test.summary for the term will have its name, DF set to NA, p-value set to NA, and action set to Absent.

See Also

asremlPlus-package, asrtests, choose.model.asrtests, reml.lrt.asreml, rmboundary.asrtests, newfit.asreml, sig.devn.reparam.asrtests, addrm.terms.asrtests

Examples

```
## Not run:
data(Wheat.dat)
current.asr <- asreml(yield ~ Rep + WithinColPairs + Variety,
                      random = ~ Row + Column + units,
                      rcov = ~ ar1(Row):ar1(Column),
                      data=Wheat.dat)
current.asrt <- asrtests(current.asr, NULL, NULL)
current.asrt <- rmboundary.asrtests(current.asrt)
# Test nugget term
current.asrt <- testranfix.asrtests("units", current.asrt, positive=TRUE)

## End(Not run)
```

`testrcov.asrtests` *Fits a new rcov formula using* `asreml` *and tests whether the change is significant.*

Description

Fits a new rcov formula using `asreml` and tests whether the change is significant. If `simpler = FALSE` the model to be fitted must be more complex than the one whose fit has been stored in `asrtests.obj`. That is, the new model must have more parameters. However, if `simpler = TRUE` the model to be fitted must be simpler than the one whose fit has been stored in `asrtests.obj` in that it must have fewer parameters. Any boundary terms are removed using `rmboundary.asrtests`, which may mean that the models are not nested. The test is a REML likelihood ratio test that is performed using `reml.lrt.asreml`, which is only valid if the models are nested. It compares the newly fitted model with the fit of the model in `asrtest.obj`. A row is added to the `test.summary` `data.frame` using the supplied `label`.

Usage

```
testrcov.asrtests(terms=NULL, asrtests.obj, label = "R model",
                  simpler = FALSE, alpha = 0.05,
                  positive.zero = FALSE, bound.test.parameters = "none",
                  denDF="default", update = TRUE, trace = FALSE,
                  set.terms = NULL, ignore.suffices = TRUE,
                  constraints = "P", initial.values = NA, ...)
```

Arguments

`terms`	a model for the `rcov` argument in `asreml`, stored as a `character`.
`asrtests.obj`	an `asrtests` object for a fitted model that is a list containing an `asreml` object, a `wald.tab` `data.frame` with 4 columns, and a `data.frame` with 5 columns that records any previous changes and tests in the fitted model.
`label`	a character string to use as the label in `test.summary` and which indicates what is being tested.

`simpler` a logical indicating whether the new model to be fitted is simpler than the already fitted model whose fit is stored in `asrtests.obj`.

`alpha` the significance level for the test.

`positive.zero` Indicates whether the hypothesized values for the variance components being tested are on the boundary of the parameter space. For example, this is true for positively-constrained variance components that, under the reduced model, are zero. This argument does not need to be set if `bound.test.parameters` is set.

`bound.test.parameters`
Indicates whether for the variance components being tested, at least some of the hypothesized values are on the boundary of the parameter space. The default is `"none"`. Other possibilities are `"onlybound"` and `"one-and-one"`. The latter signifies that there are two parameters being tested, one of which is bound and the other is not. For example, the latter is true for testing a covariance and a positively-constrained variance component that, under the reduced model, are zero.

`denDF` Specifies the method to use in computing approximate denominator degrees of freedom when `wald.asreml` is called. Can be `none` to suppress the computations, `numeric` for numerical methods, `algebraic` for algebraic methods or `default`, the default, to autommatically choose numeric or algebraic computations depending on problem size. The denominator degrees of freedom are calculated according to Kenward and Roger (1997) for fixed terms in the dense part of the model.

`update` if `TRUE` then `update.asreml` is called to fit the model with the `rcov` model supplied in `terms`. In doing this the arguments `R.param` and `G.param` are set to those in the `asreml` object stored in `asrtests.obj` so that the values from the previous model are used as starting values. If `FALSE` then a call is made to `asreml` in which the only changes from the previous call are that (i) `rcov` model is that specified in `terms` and (ii) modifications specified via `...` are made.

`trace` if TRUE then partial iteration details are displayed when ASReml-R functions are invoked; if FALSE then no output is displayed.

`set.terms` a character vector specifying the terms that are to have constraints and/or initial values set prior to fitting.

`ignore.suffices`
a logical vector specifying whether the suffices of the `asreml`-assigned names of the variance terms (i.e. the information to the right of an "!", other than "R!") is to be ignored in matching elements of `terms`. If `TRUE` for an element of `terms`, the suffices are stripped from the `asreml`-assigned names. If `FALSE` for an element of `terms`, the element must exactly match an `asreml`-assigned name for a variance term. This vector must be of length one or the same length as `terms`. If it is of length one then the same action is applied to the `asreml`-assigned suffices for all the terms in `terms`.

`constraints` a character vector specifying the constraints to be applied to the terms specified in `terms`. This vector must be of length one or the same length as `terms`. If it is of length one then the same constraint is applied to all the terms in `terms`. If any of the constraints are equal to NA then they are left unchanged for those terms.

initial.values a character vector specifying the initial values for the terms specified in terms. This vector must be of length one or the same length as terms. If it is of length one then the same initial value is applied to all the terms in terms. If any of the initial.values are equal to NA then they are left unchanged for those terms.

... further arguments passed to asreml and to wald.asreml.

Value

An asrtests object, which is a list containing:

1. asreml.obj: an asreml object containing the fit after the term has been omitted from the model;
2. wald.tab: a 4-column data.frame containing a pseudo-anova table for the fixed terms produced by wald.asreml;
3. test.summary: a data.frame with columns term, DF, denDF, p and action. A row is added to it for each term that is dropped, added or tested or a note that several terms have been added or removed. A row contains the name of the term, the DF, the p-value and the action taken. Possible codes are: Dropped, Retained, Swapped, Unswapped, Significant, Nonsignificant, Absent, Added, Removed and Boundary. If the changed model did not converge, Unconverged will be added to the code. Note that the logical asreml.obj$converge also reflects whether there is convergence.

If the term is not in the model, then the supplied asreml object will be returned. Also, reml.test will have the likelihood ratio and the p-value set to NA and the degrees of freedom to zero. Similarly, the row of test.summary for the term will have its name, a p-value set to NA, and action set to Absent.

See Also

asremlPlus-package, asrtests, newrcov.asrtests,

choose.model.asrtests, reml.lrt.asreml, rmboundary.asrtests, newfit.asreml, testswapran.asrtests, addrm.terms.asrtests, sig.devn.reparam.asrtests

Examples

```
## Not run:
data(Wheat.dat)
current.asr <- asreml(yield ~ Rep + WithinColPairs + Variety,
                      random = ~ Row + Column + units,
                      rcov = ~ ar1(Row):ar1(Column),
                      data=Wheat.dat)
current.asrt <- asrtests(current.asr, NULL, NULL)
current.asrt <- rmboundary.asrtests(current.asrt)
# Test Row autocorrelation
current.asrt <- testrcov.asrtests("~ Row:ar1(Column)", current.asrt,
                                  label="Row autocorrelation", simpler=TRUE)
print(current.asrt)

## End(Not run)
```

testswapran.asrtests *Tests, using a REMLRT, whether the difference between current random model and one in which oldterms are dropped and newterms are added is significant.*

Description

Fits a new random model using asreml by removing oldterms and adding newterms. If simpler = FALSE the model to be fitted must be more complex than the one whose fit has been stored in asrtests.obj. That is, the new model must have more parameters. However, if simpler = TRUE the model to be fitted must be simpler than the one whose fit has been stored in asrtests.obj in that it must have fewer parameters. The test is a REML ratio test that is performed using reml.lrt.asreml, which is only valid if the models are nested. It compares the newly fitted model with the fit of the model in asrtest.obj. A row is added to the test.summary data.frame using the supplied label. If the newly fitted model is retained, any boundary terms are then removed using rmboundary.asrtests.

Usage

```
testswapran.asrtests(oldterms = NULL, newterms = NULL, asrtests.obj,
                     label = "Swap in random model", simpler = FALSE, alpha = 0.05,
                       positive.zero = FALSE, bound.test.parameters = "none",
                       denDF="default", trace = FALSE, update = TRUE,
                       set.terms = NULL, ignore.suffices = TRUE,
                       constraints = "P", initial.values = NA, ...)
```

Arguments

oldterms	terms, stored as a character, that are to be removed from the random model using asreml.
newterms	terms, stored as a character, that are to be added to the random model using asreml.
asrtests.obj	an asrtests object for a fitted model that is a list containing an asreml object, a wald.tab data.frame with 4 columns, and a data.frame with 5 columns that records any previous changes and tests in the fitted model.
simpler	a logical indicating whether the new model to be fitted. after the changes made as a result of swapping oldterms for newterms, is simpler than the already fitted model whose fit is stored in asrtests.obj.
alpha	the significance level for the test.
label	a character string to use as the label in test.summary and which indicates what is being tested.
positive.zero	Indicates whether the hypothesized values for the variance components being tested are on the boundary of the parameter space. For example, this is true for positively-constrained variance components that, under the reduced model, are zero. This argument does not need to be set if bound.test.parameters is set.

`bound.test.parameters`
: Indicates whether for the variance components being tested, at least some of the hypothesized values are on the boundary of the parameter space. The default is `"none"`. Other possibilities are `"onlybound"` and `"one-and-one"`. The latter signifies that there are two parameters being tested, one of which is bound and the other is not. For example, the latter is true for testing a covariance and a positively-constrained variance component that, under the reduced model, are zero.

`denDF`
: Specifies the enthod to use in computing approximate denominator degrees of freedom when `wald.asreml` is called. Can be `none` to suppress the computations, `numeric` for numerical methods, `algebraic` for algebraic methods or `default`, the default, to autommatically choose numeric or algebraic computations depending on problem size. The denominator degrees of freedom are calculated according to Kenward and Roger (1997) for fixed terms in the dense part of the model.

`trace`
: if TRUE then partial iteration details are displayed when ASReml-R functions are invoked; if FALSE then no output is displayed.

`update`
: if `TRUE` then `update.asreml` is called to change the model. In doing this the arguments `R.param` and `G.param` are set to those in the `asreml` object stored in `asrtests.obj` so that the values from the previous model are used as starting values. If `FALSE` then a call is made to `asreml` in which the only changes from the previous call are that (i) models are modifed for the supplied `oldterms` and `newterms`, and (ii) modifications specified via `...` are made.

`set.terms`
: a character vector specifying the terms that are to have constraints and/or initial values set prior to fitting.

`ignore.suffices`
: a logical vector specifying whether the suffices of the `asreml`-assigned names of the variance terms (i.e. the information to the right of an "!", other than "R!") is to be ignored in matching elements of `terms`. If `TRUE` for an element of `terms`, the suffices are stripped from the `asreml`-assigned names. If `FALSE` for an element of `terms`, the element must exactly match an `asreml`-assigned name for a variance term. This vector must be of length one or the same length as `terms`. If it is of length one then the same action is applied to the `asreml`-assigned suffices for all the terms in `terms`.

`constraints`
: a character vector specifying the constraints to be applied to the terms specified in `terms`. This vector must be of length one or the same length as `terms`. If it is of length one then the same constraint is applied to all the terms in `terms`. If any of the constraints are equal to NA then they are left unchanged for those terms.

`initial.values`
: a character vector specifying the initial values for the terms specified in `terms`. This vector must be of length one or the same length as `terms`. If it is of length one then the same initial value is applied to all the terms in `terms`. If any of the initial.values are equal to NA then they are left unchanged for those terms.

`...`
: further arguments passed to `asreml` and `wald.asreml`.

Value

An `asrtests` object, which is a list containing:

1. asreml.obj: an asreml object containing the fit after the term has been omitted from the model;
2. wald.tab: a 4-column data.frame containing a pseudo-anova table for the fixed terms produced by wald.asreml;
3. test.summary: a data.frame with columns term, DF, denDF, p and action. A row is added to it for each term that is dropped, added or tested or a note that several terms have been added or removed. A row contains the name of the term, the DF, the p-value and the action taken. Possible codes are: Dropped, Retained, Swapped, Unswapped, Significant, Nonsignificant, Absent, Added, Removed and Boundary. If the changed model did not converge, Unconverged will be added to the code. Note that the logical asreml.obj$converge also reflects whether there is convergence.

If the term is not in the model, then the supplied asreml object will be returned. Also, reml.test will have the likelihood ratio and the p-value set to NA and the degrees of freedom to zero. Similarly, the row of test.summary for the term will have its name, a p-value set to NA, and action set to Absent.

See Also

asrtests, choose.model.asrtests, reml.lrt.asreml, rmboundary.asrtests, newfit.asreml, testrcov.asrtests, addrm.terms.asrtests, sig.devn.reparam.asrtests

Examples

```
## Not run:
  current.asrt <- testswapran.asrtests(oldterms = "str(~ Cart/xDays, ~us(2):id(184))",
                                       newterms = "Cart/xDays",
                                       current.asrt, pos = FALSE,
                                       label = "Intercept/Slope correlation",
                                       simpler = TRUE)
  print(current.asrt)

## End(Not run)
```

---

variofaces.asreml *plot empirical variogram faces, including envelopes, as described by Stefanova, Smith & Cullis (2009)*

Description

A function that produces a plot for each face of an empirical 2D variogram based on residuals produced after the fitting of a model using the function asreml. It also adds envelopes to the plot by simulating data sets from a multivariate normal distribution with expectation equal to the fitted values obtained from the fixed and spline terms and variance matrix equal to the fitted variance matrix (Stefanova, Smith & Cullis, 2009). The plot is constrolled by the rcov model, which must consist of two factors corresponding to the two physical dimensions underlying the data. It can also have a third term involving the at function that defines sections of the data, such as experiments in different environments. In this case, the two variogram faces are produced for each section.

Usage

```
variofaces.asreml(object, means=NULL, V, nsim=100, seed = NULL, tolerance=1E-10,
                  units = "ignore", update = TRUE, trace = FALSE,
                  graphics.device=NULL, ...)
```

Arguments

object	An asreml object from a call to asreml in which the data argument has been set.
means	The vector of means to be used in generating simulated data sets. If it is NULL, the fitted values based on object are used. It must be the same length as the response variable for object.
V	The fitted variance matrix, i.e. having the appropriate pattern and values given the model fitted to the observed data and the estimates of the parameters obtained.
nsim	The number of data sets to be simulated in obtaining the envelopes.
seed	A single value, interpreted as an integer, that specifies the starting value of the random number generator.
tolerance	The value such that eigenvalues less than it are consdered to be zero.
units	A character indicating whether the BLUPs for units are added to the residuals when this reserved factor is included in the random model. Possible values are addtoresiduals and ignore. If standardized conditional residuals are plotted and the BLUPs for units are to be added then it is the standardized BLUPs that are added.
update	if TRUE then the arguments R.param and G.param are set to those in the asreml object supplied in object so that the values from the original model are used as starting values. If FALSE then calls are made to asreml in which the only changes from the previous call are (i) the model is fitted to simulated data and (ii) modifications specified via ... are made, except that changes cannot be made to any of the models.
trace	if TRUE then partial iteration details are displayed when ASReml-R functions are invoked; if FALSE then no output is displayed.
graphics.device	A character specifying a graphics device for plotting. The default is graphics.device = NULL, which will result in plots being produced on the current graphics device. Setting it to "windows", for example, will result in a windows graphics device being opened.
...	Other arguments that are passed down to the function asreml. Changes to the models are not allowed. Other changes are dangerous and generally should be avoided.

Details

The rcov model is scanned to ensure that it involves only two factors not included in the at function, and to see if it has a third factor in an at function. If so, the faces of the 2D variogram, each

based on one of the two non-at factors, are derived from the residuals in the supplied asreml object using asreml.variogram, this yielding the observed variogram faces. If aom was set to TRUE for the asreml object, the standardized consitional residuals are used. Then nsim data sets are generated by adding the fitted.values, extracted from the asreml object, to a vector of values randomly generated from a normal distribution with expectation zero and variance matrix V. Each data set is analyzed using the model in object and the variogram values for the faces are obtained using asreml.variogram stored. Note, if the analysis for a data set does not converge in maxiter iterations, it is discarded and a replacement data set generated. The value of maxiter can be specified in the call to variofaces.asreml. Plots are produced for each face and include the observed values and the 2.5%, 50% & 97.5% quantiles.

Value

A list with the following components:

1. **face1:** a data.frame containing the variogram values on which the plot for the first dimension is based.
2. **face2:** a data.frame containing the variogram values on which the plot for the second dimension is based.

Author(s)

Chris Brien

References

Stefanova, K. T., Smith, A. B. & Cullis, B. R. (2009) Enhanced diagnostics for the spatial analysis of field trials. *Journal of Agricultural, Biological, and Environmental Statistics*, **14**, 392–410.

See Also

asremlPlus-package, asreml, plotvariofaces.asreml, simulate.asreml.

Examples

```
## Not run:
data(Wheat.dat)
current.asr <- asreml(yield ~ Rep + WithinColPairs + Variety,
                      random = ~ Row + Column + units,
                      rcov = ~ ar1(Row):ar1(Column),
                      data=Wheat.dat)
current.asrt <- asrtests(current.asr, NULL, NULL)
current.asrt <- rmboundary.asrtests(current.asrt)
# Form variance matrix based on estimated variance parameters
s2 <- current.asr$sigma2
gamma.Row <- current.asr$gammas[1]
gamma.unit <- current.asr$gammas[2]
rho.r <- current.asr$gammas[4]
rho.c <- current.asr$gammas[5]
row.ar1 <- mat.ar1(order=10, rho=rho.r)
col.ar1 <- mat.ar1(order=15, rho=rho.c)
```

```
V <- gamma.Row * fac.sumop(Wheat.dat$Row) +
     gamma.unit * diag(1, nrow=150, ncol=150) +
     mat.dirprod(col.ar1, row.ar1)
V <- s2*V

#Produce variogram faces plot (Stefanaova et al, 2009)
variofaces.asreml(current.asr, V=V)

## End(Not run)
```

`WaterRunoff.dat` *Data for an experiment to investigate the quality of water runoff over time*

Description

This data is from an experiment to investigate the quality of water runoff. However, it has been modified to hide the true identity of the Species and Sources. It is used to provide executable examples of the functions listed under **See Also**.

Usage

```
data(WaterRunoff.dat)
```

Format

A data.frame containing 440 observations of 13 variables.

Source

Kazemi, F. (pers. comm.)

See Also

`choose.model.asrtests`, `sig.devn.reparam.asrtests`, `predictionplot.asreml`, `predictparallel.asreml`, `pred.present.asreml`

`Wheat.dat` *Data for an experiment to investigate 25 varieties of wheat*

Description

The data appears in Gilmour et al. [1995] and is from a field experiment designed to compare the performance of 25 varieties of wheat. An analysis of it using asreml is presented by Butler et al. (2010, Sectoion 8.6), although they suggest that it is a barley experiment. It is used in `asremlPlus-package` as an executable example of the use of the `asremlPlus` to analyse a data set.

The experiment was conducted at Slate Hall Farm, UK, in 1976 and was designed as a balanced lattice square with 6 replicates laid out in a 10×15 rectangular grid. The columns in the data frame are: Rep, Row, Column, WithinColPairs, Variety, yield. The response variable is the grain yield.

Usage

```
data(Wheat.dat)
```

Format

A data.frame containing 150 observations of 6 variables.

Source

Butler, D. G., et al. (2010). *Analysis of Mixed Models for S language environments: ASReml-R reference manual.* Brisbane, DPI Publications.

Gilmour, A. R., et al. (1995) Average Information REML: An efficient algorithm for variance parameter estimation in linear mixed models. *Biometrics*, **51**, 1440-1450.

Index