Package ‘predictmeans’

February 29, 2024

Version 1.1.0
Author Dongwen Luo, Siva Ganesh and John Koolaard
Maintainer Dongwen Luo <dongwen.luo@agresearch.co.nz>
Title Predicted Means for Linear and Semiparametric Models
Description Providing functions to diagnose and make inferences from various linear models, such as those obtained from 'aov', 'lm', 'glm', 'gls', 'lme', 'lmer', 'glmmTMB' and 'semireg'. Inferences include predicted means and standard errors, contrasts, multiple comparisons, permutation tests, adjusted R-square and graphs.
Depends R (>= 3.5.0), glmmTMB, lme4, nlme, lmerTest
Imports car, ggplot2, graphics, grDevices, HRW, lmeInfo, lmeSplines, Matrix, MASS, methods, numDeriv, parallel, pbkrtest, plotly, plyr, splines2, stats, utils
License GPL (>= 2)
URL https://CRAN.R-project.org/package=predictmeans
Repository CRAN
LazyLoad yes
NeedsCompilation yes
Encoding UTF-8
ByteCompile true
Date/Publication 2024-02-29 10:30:02 UTC

R topics documented:

  predictmeans-package .................................................. 2
  ATP .............................................................................. 3
  ci_mcp ........................................................................ 3
  Clinical ......................................................................... 4
  contrastmeans .............................................................. 5
  CookD .......................................................................... 6
  covariatemeans ............................................................ 7
  df_term ......................................................................... 9

1
**predictmeans-package**

**Predicted Means for Linear and Semiparametric Models**

**Description**

This package provides functions to diagnose and make inferences from various linear models, such as those obtained from 'aov', 'lm', 'glm', 'gls', 'lme', 'lmer', 'glmmTMB' and 'semireg'. Inferences include predicted means and standard errors, contrasts, multiple comparisons, permutation tests, adjusted R-square and graphs.

**Details**

- **Package**: predictmeans
- **Type**: Package
- **Version**: 1.1.0
- **Date**: 2024-02-29
- **License**: GPL (>= 2)

**Author(s)**

Dongwen Luo, Siva Ganesh and John Koolaard

Maintainer: Dongwen Luo <dongwen.luo@agresearch.co.nz>
References

ATP
ATP containing data

Description
ATP containing data from an experiment to study the effects of preserving liquids on the enzyme content of dog hearts. There were 23 hearts and two treatment factors, A and B, each at two levels. Measurements were made of ATP as a percentage of total enzyme in the heart, at one and two hourly intervals during a twelve hour period following initial preservation.

Usage
data(ATP)

Format
ATP is a 230 row data frame with the following columns
- **heart**  dog heart id.
- **time**  time in hour for ATP measurement.
- **A**  treatment with two levels.
- **B**  treatment with two levels.
- **ATP**  percentage of total enzyme in the heart.

ci_mcp
Multiple Comparisons Based on the Confidence Intervals

Description
This function produces letter representations for a multiple comparison test by analyzing the confidence intervals associated with the mean values of different treatments. In particular, if the confidence intervals of two treatments overlap, it indicates that there is no significant difference between them. Conversely, if the confidence intervals do not overlap, it indicates that the treatments are significantly different from each other.

Usage
ci_mcp(LL, UL, trt_n=NULL)
Arguments

LL   Lower limits of treatments' confidence interval.
UL   Upper limits of treatments' confidence interval.
trt_n Treatments' names.

Author(s)

Dongwen Luo, Siva Ganesh and John Koolaard

References

Vanessa, C. (05 October 2022), Confidence tricks: the 83.4% confidence interval for comparing means, https://vsni.co.uk/blogs/confidence_trick.

Examples

library(predictmeans)
library(nlme)

data("Oats", package="nlme")
Oats$nitro <- factor(Oats$nitro)
fm <- lme(yield ~ nitro*Variety, random=~1|Block/Variety, data=Oats)
  # fm <- lmer(yield ~ nitro*Variety+(1|Block/Variety), data=Oats)
predictmeans(fm, "nitro", adj="BH", plot=FALSE)$mean_table
predictmeans(fm, "nitro", pair=TRUE, level=0.166, letterCI = TRUE, plot=FALSE)$mean_table

Clinical Clinical data

Description

Clinical data is from a multicentre randomized clinical trial (Beitler & Landis 1985, Biometrics).

Usage

data(Clinical)

Format

Clinical is a 16 row data frame with the following columns

<table>
<thead>
<tr>
<th>Clinic</th>
<th>Treatment</th>
<th>Favorable</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 centres id.</td>
<td>2 skin treatments (control or active drug).</td>
<td>number that produced a favourable response.</td>
<td>number of patients in each treatment group.</td>
</tr>
</tbody>
</table>
contrastmeans

Linear Contrast Tests for a Linear Model

Description

Performs t-tests (or permuted t-tests) of specified contrasts for linear models obtained from functions aov, lm, glm, gls, lme, or lmer.

Usage

contrastmeans(model, modelterm, ctrmatrix, ctrnames=NULL, adj="none", Df, permlist)

Arguments

model Model object returned by aov, lm, glm, gls, lme, and lmer.
modelterm Name (in "quotes") for indicating which factor term's contrast to be calculated. The modelterm must be given exactly as it appears in the printed model, e.g. "A" or "A:B".
ctrmatrix A specified contrast matrix. If ctrmatrix is missing, the program will ask user to enter it.
ctrnames Names of the specified contrasts, e.g. c("A vs D", "C vs B", ...)
adj Name (in "quote") for indicating a method for adjusting p-values of pairwise comparisons. The choices are "none", "tukey", "holm", "hochberg", "hommel", "bonferroni", "BH", "BY" and "fdr". The default method is "none".
Df A denominator degree of freedom for modelterm. (For glmer models the Df needs to be specified, while for the other models, Df is obtained from the fitted model automatically).
permlist A model parameter list containing nsim parameters produced by the function permmodels. When permlist != NULL, the option Df will be non-functional. This is a key option for the permutation test.

Value

There are two components in the output which are

Table A table showing t-test results for the specified linear contrasts.
K A contrast matrix.

Author(s)

Dongwen Luo, Siva Ganesh and John Koolaard

References

Examples

library(predictmeans)
# ftable(xtabs(yield ~ Block+Variety+nitro, data=Oats))
Oats$nitro <- factor(Oats$nitro)
fm <- lme(yield ~ nitro*Variety, random=-1|Block/Variety, data=Oats)
# library(lme4)
# fm <- lmer(yield ~ nitro*Variety+(1|Block/Variety), data=Oats)

## Not run:
## The contrast has a contrast matrix as follows:
## 0:Golden Rain 0:Marvellous 0:Victory
##[1,] -1 0 1
##[2,] 0 0 1
## 0.2:Golden Rain 0.2:Marvellous 0.2:Victory
##[1,] 0 0 0
##[2,] 0 0 0
## 0.4:Golden Rain 0.4:Marvellous 0.4:Victory
##[1,] 0 0 0
##[2,] 0 -1 0
## 0.6:Golden Rain 0.6:Marvellous 0.6:Victory
##[1,] 0 0 0
##[2,] 0 0 0

# 1. Enter above contrast matrix into a pop up window, then close the window
# contrastmeans(fm, "nitro:Variety")

# 2. Construct the contrast matrix directly
cm <- rbind(c(-1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0),
            c(0, 0, 1, 0, 0, 0, -1, 0, 0, 0, 0, 0))
contrastmeans(fm, "nitro:Variety", ctmatrix=cm)

---

CookD  
**Calculates and plots Cook’s distances for a Linear (Mixed) Model**

Description

This function produces Cook’s distance plots for a linear model obtained from functions `aov`, `lm`, `glm`, `gls`, `lme`, or `lmer`.

Usage

CookD(model, group=NULL, plot=TRUE, idn=3, newwd=TRUE)

Arguments

- **model**: Model object returned by `aov`, `lm`, `glm`, `gls`, `lme`, or `lmer`.
- **group**: Name (in “quotes”) for indicating how observations are deleted for Cook’s distance calculation. If `group!=NULL` then deletions will be along levels of `group` variable, otherwise, will be along individual observations.
`covariatemeans`  

**plot** A logical variable; if it is true, a plot of Cook’s distance will be presented. The default is TRUE.

**idn** An integer indicating the number of top Cook’s distances to be labelled in the plot. The default value is 3.

**newwd** A logical variable to indicate whether to print graph in a new window. The default value is TRUE.

**Author(s)**  

Dongwen Luo, Siva Ganesh and John Koolaard

**Examples**
```r
library(predictmeans)
Oats$nitro <- factor(Oats$nitro)
fm <- lme(yield ~ nitro*Variety, random=~1|Block/Variety, data=Oats)
# library(lme4)
# fm <- lmer(yield ~ nitro*Variety+(1|Block/Variety), data=Oats)
CookD(fm)
```

**Description**

This function obtains predicted means with graph for a new set of covariate values.

**Usage**

```r
covariatemeans(model, modelterm=NULL, covariate, as.is=FALSE, covariateV=NULL, data=NULL, level=0.05, Df=NULL, trans=NULL, transOff=0, responsen=NULL, trellis=TRUE, plotord=NULL, mtitle=NULL, ci=TRUE, point=TRUE, jitterv=0, newwd=TRUE)
```

**Arguments**

- **model** Model object returned by `aov`, `lm`, `glm`, `gls`, `lme`, and `lmer`.
- **modelterm** Name (in "quotes") for indicating which factor term’s predicted mean to be calculated. The `modelterm` must be given exactly as it appears in the printed model, e.g. "A" or "A:B".
- **covariate** Name (in "quotes") of one the covariate variables in the model.
- **as.is** A logic value to specify whether or not using original covariate values’ rage for graph, the default is FALSE.
- **covariateV** A numeric vector when `as.is` is FALSE, then `covariatemeans` will produce the result for `covariate` at value of `covariateV`.
- **data** In some cases, you need to provide the data set used in model fitting, especially when you have applied some variable transformation in the model.
level  A significant level for calculating confident interval. The default value is 0.05.

Df  A degree of freedom for calculating CI of predicted means (you can manually specified ddf here). For the above models, ddf is obtained from the function automatically.

trans  A function object for calculating the back transformed means, e.g. trans=exp.

transOff  When you use trans=exp(x+1), then transOff=1, the default is 0.

responsen  Name (in "quotes") of the back transformed response variable in the model.

trellis  A logical variable. If set to TRUE (default), a trellis plots of predicted means with CI will be drawn.

plotord  A numeric vector specifying the order of plotting for two or three way interaction (e.g. plotord = c(2, 1, 3) will put the second variable in modelterm on the X axis, the first variable as the grouping variable, and the third one as the panel variable). The defaults are c(1, 2) and c(1, 2, 3) for two and three way interactions.

mtitle  The main title in the graph.

cl  A logical variable to indicate whether to print confidence interval. The default value is TRUE.

point  A logical variable to indicate whether to print raw data points. The default value is TRUE.

jitterv  A degree of jitter in x and y direction in the graph. The default is zero.

newwd  A logical variable to indicate whether to print graph in a new window. The default value is TRUE.

Value  
Predicted Means  
A table of predicted means.

Author(s)  
Dongwen Luo, Siva Ganesh and John Koolaard

Examples  
library(predictmeans)  
data(Oats, package="nlme")  
fm <- lme(yield ~ nitro*Variety, random=~1|Block/Variety, data=Oats)  
# library(lme4)  
# fm <- lmer(yield ~ nitro*Variety+(1|Block/Variety), data=Oats)  
covariatemmeans(fm, "Variety", covariate="nitro")  
covariatemmeans(fm, "Variety", covariate="nitro", covariateV=seq(0, 0.6, 0.1))$data
df_term

Calculate the degree of freedom of a model term (contrast) for a lmer model using "Kenward-Roger" or "Satterthwaite" method.

Usage

```
df_term(model, modelterm, covariate=NULL, ctrmatrix=NULL, ctrnames=NULL, type=c("Kenward-Roger", "Satterthwaite"))
```

Arguments

- `model`: Model object returned by `lmer`.
- `modelterm`: Name (in "quotes") for indicating which factor term’s degree of freedom to be calculated. The `modelterm` must be given exactly as it appears in the model formula, e.g. "A" or "A:B".
- `covariate`: Name (in "quotes") of one the covariate variables in the model.
- `ctrmatrix`: A specified contrast matrix. If `ctrmatrix` isn’t NULL, the program will ignore `modelterm` and calculate degree of freedom for the `ctrmatrix`.
- `ctrnames`: Names of the specified contrasts, e.g. c("A vs D", "C vs B", ...)
- `type`: Name (in "quote") for indicating a method for calculating degree of freedom. The choices are "Kenward-Roger" and "Satterthwaite". The default method is "Kenward-Roger".

Author(s)

Dongwen Luo, Siva Ganesh and John Koolaard

Examples

```
library(predictmeans)
# ftable(xtabs(yield ~ Block+Variety+nitro, data=Oats))
Oats$nitro <- factor(Oats$nitro)
fm <- lmer(yield ~ nitro*Variety+(1|Block/Variety), data=Oats)
df_term(fm, "nitro:Variety")
```

## Not run:
## The contrast has a contrast matrix as follows:
## 0:Golden Rain 0:Marvellous 0:Victory
## 0.2:Golden Rain 0.2:Marvellous 0.2:Victory
```
# 0:Golden Rain 0:Marvellous 0:Victory
# [1,] -1 0 1
# [2,] 0 0 1
# 0.2:Golden Rain 0.2:Marvellous 0.2:Victory
# [1,] 0 0 0
# [2,] 0 0 0
```
# The data is for the comparison of the effectiveness of three analgesic drugs to a standard drug, morphine (Finney, Probit analysis, 3rd Edition 1971, p.103). 14 groups of mice were tested for response to the drugs at a range of doses.

## Usage

data(Drug)

## Format

Drug is a 14 row data frame with the following columns

<table>
<thead>
<tr>
<th></th>
<th>Drug type of drug.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Dose dose volume.</td>
</tr>
<tr>
<td>N</td>
<td>total number of mice in each group.</td>
</tr>
<tr>
<td>R</td>
<td>number responding mice in each group.</td>
</tr>
<tr>
<td>log10Dose</td>
<td>log10 transformed dose volume.</td>
</tr>
</tbody>
</table>
Kmatrix

Matrix of Coefficients in a Linear Model

Description

This function obtains a matrix of coefficients for parametric models such as aov, lm, glm, gls, lme, and lmer.

Usage

Kmatrix(model, modelterm, covariate=NULL, covariateV=NULL, data=NULL, prtnum=FALSE)

Arguments

- **model**: Model object returned by aov, lm, glm, gls, lme, and lmer.
- **modelterm**: Name (in "quotes") for indicating which model term’s predicted mean to be calculated. The modelterm must be given exactly as it appears in the printed model, e.g. "A" or "A:B".
- **covariate**: A numerical vector to specify values of covariates for calculating predicted means, default values are the means of the associated covariates. It also can be the name of one covariate in the model.
- **covariateV**: A numeric vector or list of numeric vector, then covariatemans will produce the result for covariate at value of covariateV.
- **data**: In some cases, you need to provide the data set used in model fitting, especially when you have applied some variable transformation in the model.
- **prtnum**: An option for printing covariate info on the screen or not. The default is FALSE.

Value

- **K**: Coefficients matrix
- **fctnames**: A model frame contains factor(s) info in the model.
- **response**: The name of response variable in the model.

Author(s)

This function heavily depends on the codes from package "lsmeans".

References

Examples

```r
library(predictmeans)
data(Oats, package="nlme")
# fm <- lmer(yield ~ nitro*Variety+(1|Block/Variety), data=Oats)
fm <- lme(yield ~ nitro*Variety, random=-1|Block/Variety, data=Oats)
Kmatrix(fm, "Variety", prtnum=TRUE)$K
Kmatrix(fm, "Variety", 0.5, prtnum=TRUE)$K
# Kmatrix(fm, "Variety", "nitro")$K
Kmatrix(fm, "Variety", "nitro", covariateV=seq(0, 0.6, 0.1))$K
```

---

### permanova.lmer

**Permutation ANOVA for lmer Model**

**Description**

This function provides permutation ANOVA for lmer model.

**Usage**

```r
permanova.lmer(model, nperm = 999, ncore=3, type = c("I", "II", "III", "1", "2", "3"), ...)
```

**Arguments**

- `model` Model object returned by lmer.
- `nperm` Number of permutation, the default value is 999.
- `ncore` Number of core for parallel computing, the default value is 3.
- `type` The type of ANOVA table requested (using SAS terminology) with Type I being the familiar sequential ANOVA table.
- `...` Use to setup option: seed – Specify a random number generator seed, for reproducible results.

**Value**

Permutation ANOVA table.

**Author(s)**

Dongwen Luo, Siva Ganesh and John Koolaard
Examples

```r
# library(predictmeans)
# Oats$nitro <- factor(Oats$nitro)
# fm <- lmer(yield ~ nitro*Variety+(1|Block/Variety), data=Oats)

## Permutation Test for model terms
# permanova.lmer(fm)
# permanova.lmer(fm, type=2)
## Compare to F test
# fm0 <- lme(yield ~ nitro*Variety, random=-1|Block/Variety, data=Oats)
# anova(fm0)
```

## Description

This function obtains permutation index for a dataset.

## Usage

```r
permindex(data, block=NULL, group=NULL, nsim=4999, seed)
```

## Arguments

- `data`: Data object used in the model fitting.
- `block`: Name (in "quotes") for the blocking factor in the data.
- `group`: Name (in "quotes") for the group factor in the data.
- `nsim`: The number of permutations. The default is 4999.
- `seed`: Specify a random number generator seed, for reproducible results.

## Value

A matrix has `nsim` columns of permuted index.

## Author(s)

Dongwen Luo, Siva Ganesh and John Koolaard

## Examples

```r
library(predictmeans)
block <- rep(1:3, each=12)
group <- rep(rep(1:3, each=4), 3)
data <- data.frame(block, group)
cbind(data, permindex(data, block="block", group="group", nsim=5))
# Permute group as a whole within each block first,
```
# then permute obs within each group.
cbind(data, permindex(data, block="block", nsim=5))
# Permute obs within each block only.
cbind(data, permindex(data, group="group", nsim=5))
# Permute groups as a whole block first,
# then permute obs within each group.
cbind(data, permindex(data, nsim=5))  # Free permutation.

---

**permlmer**

_Permutation Test of random or fixed effects for lmer model._

**Description**

This function provides permutation tests for the terms in a linear mixed model of `lmer`.

**Usage**

```r
permlmer(lmer0, lmer1, nperm = 999, ncore=3, plot=FALSE, seed)
```

**Arguments**

- `lmer0` _lmer_ model under H0, note that `lmer0` model must nest within `lmer1` model.
- `lmer1` _lmer_ model under H1, note that `lmer0` model must nest within `lmer1` model.
- `nperm` Number of permutation, the default value is 999.
- `ncore` Number of core for parallel computing, the default value is 3.
- `plot` Plot permutation distribution or not, the default value is FALSE.
- `seed` Specify a random number generator seed, for reproducible results.

**Value**

Permutation p-value.

**Author(s)**

Dongwen Luo, Siva Ganesh and John Koolaard

**References**

Examples

```r
# library(predictmeans)
## Test random effects
# fm1 <- lmer(Reaction ~ Days + (Days | Subject), sleepstudy)
# fm2 <- lmer(Reaction ~ Days + (Days || Subject), sleepstudy)
# fm3 <- update(fm1, . ~ . - (Days | Subject) + (1 | Subject))
# anova(fm1, fm2, fm3)
# permlmer(fm3, fm2)
# permlmer(fm2, fm1)

## Test fixed effects
# Oats$nitro <- factor(Oats$nitro)
# fm0 <- lmer(yield ~ nitro+Variety+(1|Block/Variety), data=Oats)
# fm <- lmer(yield ~ nitro*Variety+(1|Block/Variety), data=Oats)
# permlmer(fm0, fm)
```

permmodels

---

**Permuation Test of Linear Model**

**Description**

This function provides permutation t-tests for coefficients of (fixed) effects and permutation F-tests for the terms in a linear model such as `aov`, `lm`, `glm`, `gls`, `lme`, and `lmer`.

**Usage**

```r
permmodels(model, nperm=4999, type=c("I", "II", "III", 1, 2, 3),
            test.statistic=c("Chisq", "F", "LR", "Wald"), exact=FALSE, data=NULL,
            fo=NULL, prt=TRUE, ncore=3, seed)
```

**Arguments**

- `model` Model object returned by `aov`, `lm`, `glm`, `gls`, `lme`, and `lmer`.
- `nperm` The number of permutations. The default is 4999.
- `type` type of ANOVA test, "I", "II", "III", 1, 2, or 3. Roman numerals are equivalent to the corresponding Arabic numerals.
- `test.statistic` For type I ANOVA, F test is applied to all models, while for type II and III ANOVA, F test is performed for `lm`, Chisq test for `lm` and `gls` model, Chisq or F tests for `lmer` model and Likelihood ratio, Wald or F tests for `glm` model.
- `exact` A logical variable to indicate whether or not exact no. of permutations will be used (applicable only to free the permutation case). The default is FALSE.
- `data` In some cases, you need to provide the data set used in model fitting, especially when you have applied some variable transformation in the model.
- `fo` A model formula used in the model; fo!=NULL when the formula is specified by function formula.
prt A logical variable to indicate whether or not to print output on the screen. The default is TRUE.

ncore Number of core for parallel computing, the default value is 3.

seed Specify a random number generator seed, for reproducible results.

Value

The function produces permutation t-test table for coefficients of (fixed) effects, permutation ANOVA table for model terms and a model parameter list permlist, a list containing nsim=4999 times permutation refitted model parameters which are used in functions predictmeans and contrastmeans.

Author(s)

Dongwen Luo, Siva Ganesh and John Koolaard

Examples

```r
## Not run for simplifying process of submitting pkg to CRAN
#library(predictmeans)
#Oats$nitro <- factor(Oats$nitro)
#fm <- lme(yield ~ nitro*Variety, random=~1|Block/Variety, data=Oats)
## library(lme4)
## fm <- lmer(yield ~ nitro*Variety+(1|Block/Variety), data=Oats)
#
## Permutation Test for model terms
#system.time(
#  permlme <- permmodels(model=fm, nperm=999)
#)
#
## Permutation Test for multiple comparisons
#predictmeans(model=fm, modelterm="nitro:Variety", atvar="Variety", adj="BH",
## permlist=permlme, plot=FALSE)
#
## Permutation Test for specified contrasts
#cm <- rbind(c(-1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0),
#   c(0, 0, 1, 0, 0, 0, 0, -1, 0, 0, 0, 0))
#contrastmeans(model=fm, modelterm="nitro:Variety", ctrmatrix=cm, permlist=permlme)
```

---

**PMplot**

*Level Plot of a Matrix of p-values.*

Description

Creates a plot of p-values of pairwise comparisons.

Usage

```r
PMplot(pmatrix, level=0.05, mtitle=NULL, xlabel=NULL, margin=5, legendx=0.73, newwd=TRUE)
```
predictmeans

Arguments

- **pmatrix**: A matrix with p-values from pairwise comparisons. (This is a lower triangle matrix.)
- **level**: The level of p-value to be highlighted. Default is 0.05.
- **mtitle**: The main title in the graph.
- **xlabel**: The x and y labels in the graph.
- **margin**: A value for specifying x and y margins in the graph. The default value is 5.
- **legendx**: A value for specifying x coordinate of legend. The default value is 0.73.
- **newwd**: A logical variable to indicate whether to print graph in a new window. The default is TRUE.

Author(s)

Dongwen Luo, Siva Ganesh and John Koolaard

Examples

```r
library(predictmeans)
set.seed(2013)
pvalues <- runif(28)
pmatrix <- matrix(0,8,8)
pmatrix[lower.tri(pmatrix)] <- pvalues
round(pmatrix, 4)
PMplot(pmatrix)

Oats$nitro <- factor(Oats$nitro)
fm <- lmer(yield ~ nitro*Variety+(1|Block/Variety), data=Oats)
predictout <- predictmeans(fm, "nitro:Variety", atvar="Variety", adj="BH", barplot=TRUE)
PMplot(predictout$p_valueMatrix)
```

predictmeans

**Predicted Means of a Linear Model**

Description

This function obtains predicted means, SE of means, SED of means, LSDs and plots of means with SE bar or LSD bar for parametric models such as `aov`, `lm`, `glm`, `gls`, `lme`, and `lmer`. The function also performs pairwise comparisons and permutation tests.

Usage

```r
predictmeans(model, modelterm, data=NULL, pairwise=FALSE, atvar=NULL, adj="none", Df=NULL, lsd_bar=TRUE, level=NULL, covariate=NULL, meandecr=NULL, letterCI=FALSE, trans=I, transOff = 0, responsen=NULL, count=FALSE, plotord=NULL, lineplot=TRUE, plottitle=NULL, plotxlab=NULL, plotylab=NULL, mplot=TRUE, barplot=FALSE, pplot=TRUE, bkplot=TRUE, plot=TRUE, jitterv=0.2, basesz=12, prtnum=TRUE, prtplt=TRUE, newwd=TRUE, permlist=NULL, ncore=3, ndecimal=4)
```
predictmeans

Arguments

model  Model object returned by aov, lm, glm, gls, lme, and lmer.
modelterm  Name (in "quotes") for indicating which factor term's predicted mean to be calculated. The modelterm must be factors and given exactly as it appears in the printed model, e.g. "A" or "A:B".
data  In some cases, you need to provide the data set used in model fitting, especially when you have applied some variable transformation in the model.
pairwise  An option for showing pair-wise LSDs and p-values, or not. The default is FALSE.
atvar  When pairwise = TRUE, a quoted name indicating within levels of which variable in modelterm the multiple comparison will be performed.
adj  Name (in "quote") for indicating a method for adjusting p-values of pairwise comparisons. The choices are "none", "tukey", "holm", "hochberg", "hommel", "bonferroni", "BH", "BY" and "fdr". The default method is "none". Note that LSD can't be adjusted except for "bonferroni" method.
Df  A degree of freedom for calculating LSD. For the above models, Df is obtained from the function automatically.
lsd_bar  A logical variable to indicate to print an average LSD or SED bar on the means plot. The default is TRUE.
level  A significant level for calculating LSD, CI etc. The default value is 0.05.
covariate  A numerical vector to specify values of covariates for calculating predicted means. The default values are the means of the associated covariates.
meandecr  A logical variable to indicate whether to print letters for multiple comparisons by decreasing order of means in the mean_table. The default is NULL which indicates the mean order follows the associated factor levels.
letterCI  A logical variable to indicate printed letters for multiple comparisons by whether or not CI overlap in the mean_table. The default is FALSE.
trans  A function object for calculating the back transformed means, e.g. trans=exp.
transOff  When you use trans=exp(x+1), then transOff=1, the default is 0.
responsen  Name (in "quotes") of the back transformed response variable in the model.
count  An option for indicating the back transformed mean values are counts or not. The default is FALSE.
plotord  A numeric vector specifying the order of plotting for two or three way interaction (e.g. plotord = c(2, 1, 3) will put the second variable in modelterm on the X axis, the first variable as the grouping variable, and the third one as the panel variable). The defaults are c(1, 2) and c(1, 2, 3) for two and three way interactions.
lineplot  An option for drawing a line chart, or dot chart. The default is TRUE.
plottitle  A character vector specifying the main title for plot(s). The default is NULL.
plotxlab  A character vector specifying the x label for plot(s). The default is NULL.
plotylab  A character vector specifying the y label for plot(s). The default is NULL.
mplot  An option for drawing a means plot, or not. The default is TRUE.
predictmeans

barplot  An option for drawing a bar chart, or not. The default is FALSE.
pplot  An option for drawing a p-values plot, or not when there are more than six p-values. The default is FALSE.
bkplot  An option for drawing back transformed plot, or not. The default is TRUE.
plot  An option for drawing plots, or not. The default is TRUE.
jd  A degree of jitter in x and y direction in the back transformed means graph. The default is zero.
basesz  The base font size. The default is 12.
prtnum  An option for printing covariate information on the screen, or not. The default is TRUE.
prtplt  An option for printing plots on the screen, or not. The default is TRUE.
newwd  A logical variable to indicate whether to print graph in a new window. The default is TRUE.
permlist  A model parameter list produced by the function permmmodels. When permlist != NULL, the option Df will be non-functional. This is a key option for pairwise comparisons via permutation tests.
ncore  Number of core for parallel computing when permlist != NULL, the default value is 3.
ndecimal  An option for specifying number of decimal point to be print at predicted means table. The default is 4.

Value

Predicted Means
A table of predicted means.
Standard Error of Means
A table of standard errors of predicted means.
Standard Error of Differences
Standard errors of differences between predicted means.
LSD
Least significant differences between predicted means.
Pairwise p-value
A matrix with t-values above the diagonal and p-values below the diagonal, or matrix of pairwise comparison p-values for each level of atvar.
mean_table
A summary of predicted means result including 'Predicted means', 'Standard error', 'Df' and 'CIs'. When trans!=NULL or trans!=I, a table of back transformed means with CIs are also shown.
predictmeansPlot
ggplot of predicted means.
predictmeansBKPlot
ggplot of back transformed means.
predictmeansBarPlot
gg bar plot of predicted means.
p_valueMatrix
p_value matrix for pairwise comparison.
Note

The predictmeans function becomes confused if a factor or covariate is changed to the other in a model formula. Consequently, formulae that include calls as.factor, factor, or numeric (e.g. as.factor(income)) will cause errors. Instead, create the modified variables outside of the model formula (e.g., fincome <- as.factor(income)) and then use them in the model formula.

Factors cannot have colons in level names (e.g., "level:A"); the predictmeans function will confuse the colons with interactions; rename levels to avoid colons.

For predictmeans function, it is assumed that methods coef, vcov, model.matrix, model.frame and terms are available for model.

Author(s)

Dongwen Luo, Siva Ganesh and John Koolaard

References


Vanessa, C. (05 October 2022), Confidence tricks: the 83.4% confidence interval for comparing means, https://vsni.co.uk/blogs/confidence_trick.

Examples

```r
library(predictmeans)
ftable(xtabs(yield ~ Block+Variety+nitro, data=Oats))
Oats$nitro <- factor(Oats$nitro)
fm <- lme(yield ~ nitro*Variety, random=~1|Block/Variety, data=Oats)
# fm <- lmer(yield ~ nitro*Variety+(1|Block/Variety), data=Oats)
predictmeans(fm, "nitro", adj="BH")
predictmeans(fm, "nitro:Variety", atvar="Variety", adj="BH", line=FALSE)
predictout <- predictmeans(fm, "nitro:Variety", atvar="Variety", adj="BH", barplot=TRUE, line=FALSE)
names(predictout)
print(predictout$predictmeansPlot)
print(predictout$predictmeansBarPlot)
```

R2_glmm

An adjusted coefficient of determination (R2) for generalized linear mixed models

Description

This function produces adjusted R2 for generalized linear mixed models which was crafted following the guidance provided by Professor Hans-Peter Piepho.
Usage

\[
\text{R2\_glmm(model, over\_disp=FALSE)}
\]

Arguments

- **model**: An object returned by `lmer`, `glmer` or `glmmTMB`.
- **over\_disp**: A logical scalar to indicate whether `model` with over-dispersion or not. The default value is `FALSE`.

Value

Adjusted R2 in percentage for Total (fixed + random), Fixed, Random and individual random term.

References


Examples

```r
library(predictmeans)
Oats\$nitro <- factor(Oats\$nitro)
(fm <- lmer(yield ~ nitro*Variety+(1|Block/Variety), data=Oats))
R2\_glmm(fm)
(gm <- glmer(cbind(incidence, size - incidence) ~ period + (1 | herd),
             data = cbpp, family = binomial))
R2\_glmm(gm)
```

residplot

*Diagnostic Plots for a Linear (Mixed) Model*

Description

This function produces diagnostic plots for linear models including 'aov', 'lm', 'glm', 'gls', 'lme' and 'lmer'.

Usage

\[
\text{residplot(model, group = "none", level = 1, slope = FALSE, id = FALSE, newwd=TRUE, ask=FALSE)}
\]
Arguments

- **model**: Model object returned by `aov`, `lm`, `glm`, `gls`, `lme`, and `lmer`.
- **group**: Name (in "quotes") for indicating the variable used to show grouping in the residual vs predicted plot. If variable is a term in the model, then group will be a name of the variable such as `group="A"`, otherwise group will be the actual variable such as `group=data$A`.
- **level**: An integer 1, 2, etc. used to specify a level of the random effect for plotting. The default value is 1.
- **slope**: A logical variable. If set to TRUE, a Q-Q plot of random slope will be drawn.
- **id**: A logical variable. If set to TRUE, outliers in the residual vs fitted plot can be identified interactively.
- **newwd**: A logical variable to indicate whether to print graph in a new window. The default is TRUE.
- **ask**: logical. If TRUE (and the R session is interactive) the user is asked for input, before a new figure is drawn.

Author(s)

Dongwen Luo, Siva Ganesh and John Koolaard

Examples

```r
## Note that the order of levels of nested random effects is opposite
## between lme and lmer objects.

library(predictmeans)
Oats$nitro <- factor(Oats$nitro)
fm <- lme(yield ~ nitro*Variety, random=~1|Block/Variety, data=Oats)
residplot(fm, level=2) # lme: level=2 for random effect "Block:Variety"

# Not Run
# library(lme4)
# fm <- lmer(yield ~ nitro*Variety+(1|Block/Variety), data=Oats)
# residplot(fm) # lmer: By default level=1 for random effect "Block:Variety"
```

Description

This function produces predicted means with graph for a semi parametric model with new set of covariate values.
Usage

semipred(semireg, modelterm=NULL, covariate, sm_term=NULL, contr=NULL,
covariateV=NULL, boundary=NULL, level=0.05, trans=NULL, trellis=TRUE,
scales=c("fixed", "free", "free_x", "free_y"),
plotord=NULL, ci=TRUE, point=TRUE, jitterv=0, threeD=FALSE, prt=TRUE)

Arguments

semireg  A list object returned by semireg.
modelterm  Name (in "quotes") for indicating which factor term's predicted mean to be calculated. The modelterm must be given exactly as it appears in semireg model, e.g. "A" or "A:B". In case modelterm is the same as covariate or NULL, then semipred will produce predictmeans with CI based on covariate only with out any grouping.
covariate  Name (in "quotes") of one or two (for Ztps smooth) the covariate variables in the semireg such as "x1" or c("x1", "x2").
sm_term  Names (in "quotes") of smooth terms (from smoothZ list in semireg model) used in the prediction such as "sm1_grp" or c("sm1_grp", "sm2_grp"). The default is using all smooth terms which is sm_term=NULL.
contr  A numeric vector with length of two (e.g. c(4, 1)) which indicates to produce predicted mean with CI for difference between modelterm level 4 vs level 1 along covariate.
covariateV  A numeric vector or matrix, then semipred will produce the result for covariate at value of covariateV.
boundary  A matrix or data frame of two columns, used to specify boundary of longitude and latitude, it is functional when the length of covariate is two.
level  A significant level for calculating confident interval. The default value is 0.05.
trans  A function object for calculating the back transformed means, e.g. trans=exp.
trellis  A logical scalar. If set to TRUE (default), a trellis plots of predicted means with CI will be drawn.
scales  Should scales be fixed ("fixed", the default), free ("free"), or free in one dimension ("free_x", "free_y") in a trellis graph?
plotord  A numeric vector specifying the order of plotting for two way interaction (e.g. plotord = c(2, 1) will put the second variable in modelterm on the X axis, the first variable as the grouping variable, and the third one as the panel variable). The defaults are c(1, 2) for two way interactions.
ci  A logical scalar to indicate whether to print confidence interval. The default value is TRUE.
point  A logical scalar to indicate whether to print raw data points. The default value is TRUE.
jitterv  A degree of jitter in x and y direction in the graph. The default is zero.
threeD  A logical scalar to indicate whether to produce a 3-D plot or not. The default value is FALSE.
prt  A logical scalar to indicate whether to produce plots on the screen or not. The default value is TRUE.
Value

plt       A ggplot object.
pred_df   A data.frame with predicted data.

Author(s)

Dongwen Luo, Siva Ganesh and John Koolaard

Examples

# library(predictmeans)
# data(Dialyzer, package="nlme")
# help(Dialyzer)
# str(Dialyzer)
#
# library(ggplot2)
# ggplot(Dialyzer, aes(x=rate, y=pressure, col=QB)) +
# geom_line() +
# facet_wrap(vars(Subject))
#
# fm <- semireg(pressure ~ rate*QB+(rate|Subject),
# smoothZ=list(
#     qb_grp=smZ(rate, by=QB, group=TRUE)
# ),
#     data=Dialyzer)
# str(fm$data)
# summary(fm$semer)
# residplot(fm$semer)
# anova(fm$semer)
# ranova(fm$semer)
#
# ap_out1 <- semipred(fm, "QB", "rate")
# str(ap_out1$pred_df)
# ap_out2 <- semipred(fm, "QB", "rate", contr=c(1,2))
# str(ap_out2$pred_df)
#
# help(sleepstudy)
# str(sleepstudy)
# library(latticeExtra)
# x11()
# xyplot(Reaction ~ Days | Subject, sleepstudy, aspect = "xy",
#     layout = c(9, 2), type = c("g", "p", "r"),
#     index.cond = function(x, y) coef(lm(y ~ x))[2],
#     xlab = "Days of sleep deprivation",
#     ylab = "Average reaction time (ms)",
#     as.table = TRUE)
#
# sleep.semi <- semireg(Reaction ~ Days*Subject,
#     smoothZ=list(
#         sub_grp=smZ(Days, by=Subject, group=TRUE)
#     ),
#     data=sleepstudy)
semireg

Fitting Semi Parametric Models Using lme4 Ecosystem

Description

Fit a semi parametric model based on lme4 ecosystem including lmer, glmer and glmer.nb.

Usage

semireg(formula, data, family = NULL, ngbinomial=FALSE, REML = TRUE,
        smoothZ = list(), ncenter=TRUE, nscale=FALSE, resp_scale=FALSE,
        control = lmerControl(optimizer="bobyqa"), start = NULL,
        verbose = FALSE, drop.unused.levels=TRUE, subset, weights,
        offset, contrasts = NULL, prt=TRUE, predict_info=TRUE, ...)

Arguments

formula A two-sided linear formula object describing both the fixed-effects and random-
effects part of the model, with the response on the left of a ~ operator and the
terms, separated by + operators, on the right. Random-effects terms are distin-
guished by vertical bars ("|") separating expressions for design matrices from
 grouping factors.

data A data frame or list containing the model response variable and covariates re-
quired by the formula. By default the variables are taken from environment(formula
and smoothZ), typically the environment from which semireg is called.

family A GLM family, see glm and family.

ngbinomial Logical scalar - Should a negative binomial GLMMs be used?.

REML Logical scalar - Should the estimates be chosen to optimize the REML criterion
(as opposed to the log-likelihood)?

smoothZ A list includes a set of smooth Z matrices (called 'smooth term') used in the
mixed effects model, the name of 'smooth term' should be different any vari-
ables in the model, each 'smooth term' is the result of function smZ. e.g. smoothZ=list(sm1=smZ(x1),
  sm2=smZ(x2, by=f1), sm3=smZ(x3, by=f2, group=TRUE), ...) where 'sm1' to
 'sm3' should be new variable names in the data, and x1 to x3 are covariates,
 and f1, f2 are factors.

# residplot(sleep.semi$semer)
# summary(sleep.semi$semer)
# anova(sleep.semi$semer)
# ranova(sleep.semi$semer)

# x11()
# predout1 <- semipred(sleep.semi, "Subject", "Days")
# str(predout1$pred_df)
# x11()
# predout2 <- semipred(sleep.semi, "Subject", "Days", contr = c(6,1))
# str(predout2$pred_df)
ncenter Logical scalar - Should the numeric predictors to be centered or not?
nscale Logical scalar - Should the numeric predictors to be scaled or not?
resp_scale Logical scalar - Should the response be involved in the scaling action or not?
control A list (of correct class, resulting from lmerControl() or glmerControl() respectively) containing control parameters, including the nonlinear optimizer to be used and parameters to be passed through to the nonlinear optimizer, see the *lmerControl documentation for details.
start Starting value list as used by lmer or glmer.
verbose Passed on to fitting lme4 fitting routines.
drop.unused.levels By default unused levels are dropped from factors before fitting. For some smooths involving factor variables you might want to turn this off. Only do so if you know what you are doing.
subset An optional expression indicating the subset of the rows of data that should be used in the fit. This can be a logical vector, or a numeric vector indicating which observation numbers are to be included, or a character vector of the row names to be included. All observations are included by default.
weights An optional vector of ‘prior weights’ to be used in the fitting process. Should be NULL or a numeric vector.
offset This can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases. One or more offset terms can be included in the formula instead or as well, and if more than one is specified their sum is used. See model.offset.
contrasts An optional list. See the contrasts.arg of model.matrix.default.
prt Logical scalar - Should the info to be print on screen in the middle of the process or not?
predict_info Logical scalar - Should provide the info for function semipred or not?
... Further arguments for passing on to model setup routines.

Details
A semi parametric model can be parameterized as a linear (or generalized linear) mixed model in which its random effects are smooth functions of some covariates (named ‘smooth term’). semireg follows the approach suggested by Wand and Ormerod (2008) and represents the ‘smooth term’ using O’Sullivan-type of Z.

Value
semer A mer model used in the fitting.
data A data.frame with generated variables in the fitting.
formul_vars Name of variables in the formula of semireg model.
sm_vars Name of variables in the smoothZ list.
smoothZ_call A call used to produce smooth terms in the fitting.
semireg

knots_lst  Knots used in each smooth term in the fitting.
range_lst  Range of covariate used in each smooth term in the fitting.
cov_lst    Covariance matrix list for each smooth term.
u_lst      Random effects list for each smooth term.
type_lst   Smooth type list of smooth terms.
CovMat     Covariance matrix for all smooth terms.
Cov_ind    Covariance matrix index for each smooth term.
Cov_indN   Covariance matrix index for each smooth term when group=TRUE in smoothZ argument.
df         Degree of freedom of all random terms.
lmerc      Call used in the mer model in the fitting.

Author(s)

Dongwen Luo, Siva Ganesh and John Koolaard

References


Examples

```r
## Not run
# library(predictmeans)
# library(HRW)
# data(WarsawApts)
# help(WarsawApts)
# str(WarsawApts)
# fit1 <- semireg(areaPerMzloty ~ construction.date, 
# smoothZ=list(
#     grp=smZ(construction.date, k=25)
# ),
# data = WarsawApts)
# sp_out1 <- semipred(fit1, "construction.date", "construction.date")
# WarsawApts$district <- factor(WarsawApts$district)
# fit2 <- semireg(areaPerMzloty ~ construction.date*district, resp_scale = TRUE, 
# smoothZ=list(group=smZ(construction.date, k=15, 
# by = district, group=TRUE)),
# data=WarsawApts)
# sp_out2_1 <- semipred(fit2, "district", "construction.date")
# sp_out2_2 <- semipred(fit2, "district", "construction.date", contr=c(2,1))
# data(indonRespir)
# help(indonRespir)
# str(indonRespir)
# fit3 <- semireg(respirInfection ~ age+vitAdefic + female + height
#     + stunted + visit2 + visit3 + visit4 + visit5 + visit6+(1|idnum),
```
semireg_tmb

Fitting Semi Parametric Models Using glmmTMB

Description

Fit a semi parametric model based on glmmTMB.

Usage

semireg_tmb(formula, data, family = gaussian(), smoothZ = list(), ziformula = ~0, dispformula = ~1, weights = NULL, offset = NULL, contrasts = NULL, na.action, se = TRUE, verbose = FALSE, doFit = TRUE, control = glmmTMBControl(), REML = TRUE, start = NULL, map = NULL, sparseX = NULL, prt=TRUE, predict_info=TRUE)

Arguments

formula A two-sided linear formula object describing both the fixed-effects and random-effects part of the model, with the response on the left of a ~ operator and the terms, separated by + operators, on the right. Random-effects terms are distinguished by vertical bars ("|") separating expressions for design matrices from grouping factors.

data data frame (tibbles are OK) containing model variables. Not required, but strongly recommended; if data is not specified, downstream methods such as prediction with new data (predict(fitted_model, newdata = ...)) will fail. If it is necessary to call glmmTMB with model variables taken from the environment rather than from a data frame, specifying data=NULL will suppress the warning message.

family a family function, a character string naming a family function, or the result of a call to a family function (variance/link function) information. See family for a generic discussion of families or family_glmmTMB for details of glmmTMB-specific families.

smoothZ A list includes a set of smooth Z matrixs (called 'smooth term') used in the mixed effects model, the name of 'smooth term' should be different any variables in the model, each 'smooth term' is the result of function smZ, e.g. smoothZ=list(sm1=smZ(x1),.....)
sm2=smZ(x2, by=f1), sm3=smZ(x3, by=f2, group=TRUE), ...) where 'sm1' to 'sm3' should be new variable names in the data, and x1 to x3 are covariates, and f1, f2 are factors.

**ziformula**

A one-sided (i.e., no response variable) formula for zero-inflation combining fixed and random effects: the default ~0 specifies no zero-inflation. Specifying ~. sets the zero-inflation formula identical to the right-hand side of formula (i.e., the conditional effects formula); terms can also be added or subtracted. **When using ~. as the zero-inflation formula in models where the conditional effects formula contains an offset term, the offset term will automatically be dropped.** The zero-inflation model uses a logit link.

**dispformula**

A one-sided formula for dispersion containing only fixed effects: the default ~1 specifies the standard dispersion given any family. The argument is ignored for families that do not have a dispersion parameter. For an explanation of the dispersion parameter for each family, see sigma. The dispersion model uses a log link. In Gaussian mixed models, dispformula=~0 fixes the residual variance to be 0 (actually a small non-zero value), forcing variance into the random effects. The precise value can be controlled via control=glmmTMBControl(zero_dispscale=...) the default value is sqrt(.Machine$double.eps).

**weights**

Weights, as in glm. Not automatically scaled to have sum 1.

**offset**

Offset for conditional model (only).

**contrasts**

An optional list, e.g., list(fac1="contr.sum"). See the contrasts.arg of model.matrix.default.

**na.action**

A function that specifies how to handle observations containing NAs. The default action (na.omit, inherited from the ‘factory fresh’ value of getOption("na.action")) strips any observations with any missing values in any variables. Using na.action = na.exclude will similarly drop observations with missing values while fitting the model, but will fill in NA values for the predicted and residual values for cases that were excluded during the fitting process because of missingness.

**se**

Whether to return standard errors.

**verbose**

Whether progress indication should be printed to the console.

**doFit**

Whether to fit the full model, or (if FALSE) return the preprocessed data and parameter objects, without fitting the model.

**control**

Control parameters, see glmmTMBControl.

**REML**

Whether to use REML estimation rather than maximum likelihood.

**start**

Starting values, expressed as a list with possible components beta, betazi, betad (fixed-effect parameters for conditional, zero-inflation, dispersion models); b, bzi (conditional modes for conditional and zero-inflation models); theta, thetazi (random-effect parameters, on the standard deviation/Cholesky scale, for conditional and z-i models); psi (extra family parameters, e.g., shape for Tweedie models).

**map**

A list specifying which parameter values should be fixed to a constant value rather than estimated. Map should be a named list containing factors corresponding to a subset of the internal parameter names (see start parameter). Distinct factor values are fitted as separate parameter values, NA values are held fixed:
e.g., map=list(beta=factor(c(1,2,3,NA))) would fit the first three fixed-effect parameters of the conditional model and fix the fourth parameter to its starting value. In general, users will probably want to use start to specify non-default starting values for fixed parameters. See MakeADFun for more details.

sparseX

A named logical vector containing (possibly) elements named "cond", "zi", "disp" to indicate whether fixed-effect model matrices for particular model components should be generated as sparse matrices, e.g. c(cond=TRUE). Default is all FALSE.

prt

Logical scalar - Should the info to be print on screen in the middle of the process or not?

predict_info

Logical scalar - Should provide the info for function semipred or not? In case of there is a correlation theta parameter appearing, you may set predict=FALSE.

Details

A semi parametric model can be parameterized as a linear (or generalized linear) mixed model in which its random effects are smooth functions of some covariates (named ‘smooth term’). semireg_tmb follows the approach suggested by Wand and Ormerod (2008) and represents the ‘smooth term’ using O’Sullivan-type of Z.

Value

semer A glmmTMB model used in the fitting.
data A data.frame with generated variables in the fitting.
formul_vars Name of variables in the formula of semireg_tmb model.
sm_vars Name of variables in the smoothZ list.
smoothZ_call A call used to produce smooth terms in the fitting.
knots_lst Knots used in each smooth term in the fitting.
range_lst Range of covariate used in each smooth term in the fitting.
cov_lst Covariance matrix list for each smooth term.
u_lst Random effects list for each smooth term.
type_lst Smooth type list of smooth terms.
CovMat Covariance matrix for all smooth terms.
Cov_ind Covariance matrix index for each smooth term.
Cov_indN Covariance matrix index for each smooth term when group=TRUE in smoothZ argument.
df Degree of freedom of all random terms.
tmbf The glmmTMB model result using doFit=FALSE.

Author(s)

Dongwen Luo, Siva Ganesh and John Koolaard
se_ranef

References


Examples

```r
## Not run
# library(predictmeans)
# library(HRW)
# data(WarsawApts)
# help(WarsawApts)
# str(WarsawApts)
# fit1 <- semireg_tmb(areaPerMzloty ~ construction.date,
# smoothZ=list(
#   grp=smZ(construction.date, k=25)
# ),
# data = WarsawApts)
# sp_out1 <- semipred(fit1, "construction.date", "construction.date")
# WarsawApts$district <- factor(WarsawApts$district)
# fit2 <- semireg_tmb(areaPerMzloty ~ construction.date*district, resp_scale = TRUE,
# smoothZ=list(group=smZ(construction.date, k=15,
#   by = district, group=TRUE)),
# data=WarsawApts)
# sp_out2_1 <- semipred(fit2, "district", "construction.date")
# sp_out2_2 <- semipred(fit2, "district", "construction.date", contr=c(2,1))
# data(indonRespir)
# help(indonRespir)
# str(indonRespir)
# fit3 <- semireg_tmb(respirInfec ~ age+vitAdefic + female + height
#   + stunted + visit2 + visit3 + visit4 + visit5 + visit6+(1|idnum),
# smoothZ=list(
#   grp=smZ(age)
# ),
# family = binomial,
# data = indonRespir)
# sp_out3 <- semipred(fit3, "age", "age")
# library(ggplot2)
# sp_out3$plt+
# geom_rug(data = subset(indonRespir, respirInfec==0), sides = "b", col="deeppink") +
# geom_rug(data = subset(indonRespir, respirInfec==1), sides = "t", col="deeppink")+
# ylim(0, 0.2)
```

se_ranef

Extract Standard Errors of Model Random Effects

Description

These functions extract standard errors of model random effects from objects returned by modeling functions.
Usage

se_ranef(object, rand_term=NULL)

Arguments

object object of `merMod` and `glmmTMB` fit
rand_term a name of random term in the model.

Details

se_ranef extracts standard errors of the random effects from objects returned by `lmer`, `glmer` and `glmmTMB` functions.

Value

se_ranef gives a list of standard errors for `ranef`.

Author(s)

Dongwen Luo, Siva Ganesh and John Koolaard

References

This function is modified from function 'se.ranef' at package 'arm'.

---

smZ Generate Sparse Matrix Z for penalized spline smoothing

Description

Constructs a sparse matrix (Z) of a spline function with for a covariate with(out) group.

Usage

smZ(x, k=5, intKnots=NULL, range.x=NULL, degree=3,
type=c("ZOull", "Ztps", "ns", "bs", "bernstein", "bSpline",
"nSpline", "cSpline", "iSpline", "mSpline", "smspline"),
by=NULL, group=FALSE, intercept=FALSE, pred=FALSE, ...)

Arguments

x x covariate for the smooth function. Missing values are allowed and will be returned as they are.

k Degree of freedom that equals to the column number of the returned matrix. One can specify df rather than knots, then the function chooses df - degree - as.integer(intercept) internal knots at suitable quantiles of x ignoring missing values and those x outside of the boundary. If internal knots are specified via knots, the specified df will be ignored.
intKnots: Ordered array of length smaller than that of x and consisting of unique numbers between min(x) and max(x) that specifies the positions of internal knots, that define the spline basis (see the Wand and Ormerod (2008) reference below for full mathematical details).

range.x: Array of length 2 such that range.x[1] >= min(x) and range.x[2] <= max(x).

degree: Integer: degree of (truncated) polynomial.

type: Type of splines including "ZOSull", "Ztps", "ns", "bs", "bernstein", "bSpline", "nSpline", "cSpline", "iSpline", "mSpline" and "smspline", the default is "ZO-Sull".

by: Factor for group wise splines.

group: When by != NULL, producing group wise splines with random effects separately.

intercept: If TRUE, all of the spline basis functions are returned. Notice that when using I-Spline for monotonic regression, intercept = TRUE should be set even when an intercept term is considered additional to the spline basis functions.

pred: If TRUE, the function `smZ` will be applied for prediction purpose, this option mainly used by function `semipred` internally.

Value

Z: A (or a list of) spline design matrix used in the list `smoothZ`.

Author(s)

Dongwen Luo, Siva Ganesh and John Koolaard

References


Examples

```r
x <- seq.int(0, 1, by = 0.01)
knots <- c(0.3, 0.5, 0.6)

zosuMat <- smZ(x, intKnots = knots)
bsMat <- smZ(x, intKnots = knots, degree = 2, type="bs")
isMat <- smZ(x, intKnots = knots, degree = 2, type="iSpline")

splst <- list(zosuMat, bsMat, isMat)
for (i in splst) {
  op <- par(mar = c(2.5, 2.5, 0.2, 0.1), mgp = c(1.5, 0.5, 0))
  matplot(x, i, type = "l", ylab = "I-spline basis")
  abline(v = knots, lty = 2, col = "gray")
```
### reset to previous plotting settings
```
par(op)
```
}

f <- gl(4, 25, length=length(x))
zosuMat_by <- smZ(x, intKnots = knots, by=f) # one sparse matrix
str(zosuMat_by)

zosuMat_by <- smZ(x, intKnots = knots, by=f, group=TRUE) # a list of sparse matrix
str(zosuMat_by)

---

varcomp

*varcomp*  
*Calculate SE and CI of variance components for lmer, glmer, lme, glmmTMB model*

**Description**

This function calculates SE and CI of variance components for lmer, glmer, lme, glmmTMB model.

**Usage**

```r
varcomp(model, ci=TRUE, level=0.95)
```

**Arguments**

- `model`  
  Model object returned by lmer, glmer, lme, glmmTMB.

- `ci`  
  a logical value to indicates wheather or not to simulate a confidence interval for lmer model, the default value is TRUE.

- `level`  
  level of confidence of CI, the default value is 0.95.

**Value**

Variance components table.

**Author(s)**

Dongwen Luo, Siva Ganesh and John Koolaard

**Examples**

```r
library(predictmeans)
Oats$nitro <- factor(Oats$nitro)
fm <- lmer(yield ~ nitro*Variety+(1|Block/Variety), data=Oats)
## Not run: varcomp(fm)
fm1 <- lme(yield ~ nitro*Variety, random=~1|Block/Variety, data=Oats)
varcomp(fm1)
data(Orthodont, package="nlme")
mod <- lmer(distance ~ age + (age|Subject), data=Orthodont)
```
## Not run: varcomp(mod)

mod1 <- lme(distance ~ age, random=~age|Subject, data=Orthodont)
varcomp(mod1)
Index

* package
  predictmeans-package, 2

ATP, 3

ci_mcp, 3
Clinical, 4
contrastmeans, 5
CookD, 6
covariatemeans, 7
df_term, 9
Drug, 10

family, 28
family_glmTMB, 28

glmTMBControl, 29

Kmatrix, 11

MakeADFun, 30
model.matrix.default, 29

permanova.lmer, 12
permindex, 13
permlmer, 14
permmodels, 15
PMplot, 16
predictmeans, 17
predictmeans-package, 2

R2_glmm, 20
residplot, 21

se_ranef, 31
semipred, 22
semireg, 25
semireg_tmb, 28
sigma, 29
smZ, 32

varcomp, 34