The dominanceanalysis package allows to perform the dominance analysis for multiple regression models, such as OLS (univariate and multivariate), GLM and HLM. The dominance analysis on this package is performed by `dominanceAnalysis` function. To perform bootstrap procedures you should use `bootDominanceAnalysis` function. For both, standard print and summary functions are provided.

Main Features

- Provides complete, conditional and general dominance analysis for lm (univariate and multivariate), lmer and glm (family=binomial) models.
- Covariance / correlation matrixes could be used as input for OLS dominance analysis, using `lmWithCov` and `mlmWithCov` methods, respectively.
- Multiple criteria can be used as fit indices, which is useful especially for HLM.
About Dominance Analysis

Dominance analysis is a method developed to evaluate the importance of each predictor in the selected regression model: "one predictor is 'more important than another' if it contributes more to the prediction of the criterion than does its competitor at a given level of analysis." (Azen & Budescu, 2003, p.133).

The original method was developed for OLS regression (Budescu, 1993). Later, several definitions of dominance and bootstrap procedures were provided by Azen & Budescu (2003), as well as adaptations to Generalized Linear Models (Azen & Traxel, 2009) and Hierarchical Linear Models (Luo & Azen, 2013).

Author(s)

Claudio Bustos <clbustos_at_gmail_com>, Filipa Coutinho Soares (documentation)

References


See Also

dominanceAnalysis, bootDominanceAnalysis

Examples

# Basic dominance analysis

data(longley)
lm.1<-lm(Employed~.,longley)
da<-dominanceAnalysis(lm.1)
print(da)
summary(da)
plot(da,which.graph='complete')
plot(da,which.graph='conditional')
plot(da,which.graph='general')
averageContribution

Retrieve average contribution of each predictor in a dominance analysis.

Description

Retrieve the average contribution for each predictor. Is calculated averaging all contribution by level. The average contribution defines general dominance.

Usage

averageContribution(da.object, fit.functions = NULL)
Arguments

- **da.object**: dominanceAnalysis object
- **fit.functions**: name of the fit indices to retrieve. If NULL, all fit indices will be retrieved

Value

- a list. Key corresponds to fit-index and the value is vector, with average contribution for each variable

See Also

Other retrieval methods: `contributionByLevel()`, `dominanceBriefing()`, `dominanceMatrix()`, `getFits()`

Examples

```r
data(longley)
da.longley <- dominanceAnalysis(lm(Employed~.,longley))
averageContribution(da.longley)
```

---

**bootAverageDominanceAnalysis**

*Bootstrap average values for Dominance Analysis*

Description

Bootstrap average values and correspond standard errors for each predictor in the dominance analysis. Those values are used for general dominance.

Usage

```r
bootAverageDominanceAnalysis(
  object,
  R,
  constants = c(),
  terms = NULL,
  fit.functions = "default",
  null.model = NULL,
  ...
)
```

Arguments

- **object**: lm, glm or lmer model
- **R**: number on bootstrap resamples
- **constants**: vector of predictors to remain unchanged between models. i.e. vector of variables not subjected to bootstrap analysis.
terms vector of terms to be analyzed. By default, obtained from the model
fit.functions list of functions which provides fit indices for model. See fit.functions param in dominanceAnalysis function.
null.model only for linear mixed models, null model against to test the submodels. i.e. only random effects, without any fixed effect.
... Other arguments provided to lm or lmer (not implemented yet).

Details
Use summary() to get a nice formatted data.frame object.

Examples

```r
lm.1<-lm(Employed~.,longley)
da.ave.boot<-bootAverageDominanceAnalysis(lm.1,R=1000)
summary(da.ave.boot)
```

bootDominanceAnalysis Bootstrap analysis for Dominance Analysis

Description
Bootstrap procedure as presented on Azen and Budescu (2003). Provides the expected level of dominance of predictor $X_i$ over $X_j$, as the degree to which the pattern found on sample is reproduced on the bootstrap samples. Use summary() to get a nice formatted data.frame

Usage

```r
bootDominanceAnalysis(
  x,
  R,
  constants = c(),
  terms = NULL,
  fit.functions = "default",
  null.model = NULL,
  ...
)
```

Arguments

- **x**: lm, glm or lmer model
- **R**: number on bootstrap resamples
- **constants**: vector of predictors to remain unchanged between models. i.e. vector of variables not subjected to bootstrap analysis.
contributionByLevel

Terms

- **terms**: vector of terms to be analyzed. By default, obtained from the model.
- **fit.functions**: list of functions which provides fit indices for model. See fit.functions param in dominanceAnalysis function.
- **null.model**: only for linear mixed models, null model against to test the submodels. i.e. only random effects, without any fixed effect.
- **...**: Other arguments provided to lm or lmer (not implemented yet).

Examples

```r
lm.1<-lm(Employed~.,longley)
da.boot<-bootDominanceAnalysis(lm.1,R=1000)
summary(da.boot)

contributionByLevel(da.longley)
```

---

**contributionByLevel**

*Retrieve average contribution by level for each predictor*

**Description**

Retrieve the average contribution by level for each predictor in a dominance analysis. The average contribution defines conditional dominance.

**Usage**

```r
contributionByLevel(da.object, fit.functions = NULL)
```

**Arguments**

- **da.object**: dominanceAnalysis object
- **fit.functions**: name of the fit indices to retrieve. If NULL, all fit indices will be retrieved

**Value**

a list. Key corresponds to fit-index and the value is a matrix, with contribution of each variable by level

**See Also**

Other retrieval methods: `averageContribution()`, `dominanceBriefing()`, `dominanceMatrix()`, `getFits()`

**Examples**

```r
data(longley)
da.longley<-dominanceAnalysis(lm(Employed~.,longley))
contributionByLevel(da.longley)
```
da.betareg.fit  

Provides fit indices for betareg models.

Description

Nagelkerke and Estrella are not provided because are designed for discrete dependent variables. Cox and Snell is preferred and pseudo-$R^2$ should be preferred, because McFadden’s index could be negative.

Usage

```
das.betareg.fit(data, link.betareg, ...)
```

Arguments

- `data`: complete data set
- `link.betareg`: link function for the mean model. By default, logit.
- `...`: ignored

Value

A function described by using-fit-indices. You could retrieve following indices:

- `r2.pseudo`: Provided by betareg by default
- `r2.m`: McFadden(1974)

References


See Also

Other fit indices: `da.dynlm.fit()`, `da.glm.fit()`, `da.lm.fit()`, `da.lmWithCov.fit()`, `da.lmerMod.fit()`, `da.mlmWithCov.fit()`
\textit{da.dynlm.fit} \textit{Provides coefficient of determination for dynlm models.}

\textbf{Description}

Uses $R^2$ (coefficient of determination) as fit index

\textbf{Usage}

\texttt{da.dynlm.fit(data, \ldots)}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{data} \hspace{1cm} complete data set containing the variables in the model
  \item \texttt{\ldots} \hspace{1cm} ignored
\end{itemize}

\textbf{Value}

A function described by using-fit-indices description for interface

\textbf{See Also}

Other fit indices: \texttt{da.betareg.fit()}, \texttt{da.glm.fit()}, \texttt{da.lm.fit()}, \texttt{da.lmWithCov.fit()}, \texttt{da.lmerMod.fit()}, \texttt{da.mlmWithCov.fit()}

\textit{da.glm.fit} \textit{Provides fit indices for GLM models.}

\textbf{Description}

Functions only available for logistic regression, based on Azen and Traxel(2009).

\textbf{Usage}

\texttt{da.glm.fit(data, family.glm, \ldots)}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{data} \hspace{1cm} complete data set
  \item \texttt{family.glm} \hspace{1cm} family for glm method. Use 'binomial' for logistic regression.
  \item \texttt{\ldots} \hspace{1cm} ignored
\end{itemize}

\textbf{Details}

Check \texttt{daRawResults}.
Value
A function described by using-fit-indices. You could retrieve following indices

- \( r^2.m \) McFadden(1974)
- \( r^2.cs \) Cox and Snell(1989). Use with caution, because don’t have 1 as upper bound
- \( r^2.n \) Nagelkerke(1991), that corrects the upper bound of Cox and Snell(1989) index
- \( r^2.e \) Estrella(1998)

References

See Also
Other fit indices: `da.betareg.fit()`, `da.dynlm.fit()`, `da.lm.fit()`, `da.lmWithCov.fit()`, `da.lmerMod.fit()`, `da.mlmWithCov.fit()`

Examples
```r
x1<-rnorm(1000)
x2<-rnorm(1000)
x3<-rnorm(1000)
y<-factor(runif(1000) > exp(x1+x2+x3)/(1+exp(x1+x2+x3)))
df.1=data.frame(x1,x2,x3,y)
da.glm.fit(data=df.1)("names")
da.glm.fit(data=df.1, family=glm('binomial')(y~x1)
```

---

`da.lm.fit` provides coefficient of determination for `lm` models.

Description
Uses \( R^2 \) (coefficient of determination) as fit index

Usage
`da.lm.fit(data, ...)`
Arguments

data complete data set containing the variables in the model
... ignored

Value

A function described by using-fit-indices description for interface

See Also

Other fit indices: da.betareg.fit(), da.dynlm.fit(), da.glm.fit(), da.lmWithCov.fit(),
da.lmerMod.fit(), da.mlmWithCov.fit()

Examples

x1<-rnorm(1000)
x2<-rnorm(1000)
y <-x1+x2+runif(1000)
df.1=data.frame(y=y,x1=x1,x2=x2)
da.lm.fit(df.1)("names")
da.lm.fit(df.1)(y=x1)

dalmerMod.fit Provides fit indices for hierarchical linear models, based on Luo and
Azen (2013).

Description

Provides fit indices for hierarchical linear models, based on Luo and Azen (2013).

Usage

da.lmerMod.fit(data, null.model, ...)

Arguments

data complete data set containing the variables in the model
null.model needed for HLM models
... ignored

References

  doi:10.3102/1076998612458319
See Also

Other fit indices: da.betareg.fit(), da.dynlm.fit(), da.glm.fit(), da.lm.fit(), da.lmWithCov.fit(), da.mlmWithCov.fit()

\[ R^2 \] (coefficient of determination) See lMWithCov

Usage

da.lmWithCov.fit(base.cov, ...)

Arguments

base.cov variance/covariance matrix
...

See Also

Other fit indices: da.betareg.fit(), da.dynlm.fit(), da.glm.fit(), da.lm.fit(), da.lmerMod.fit(), da.mlmWithCov.fit()

Provides coefficient of determination for multivariate models.

Usage

da.mlmWithCov.fit(base.cov, ...)

Arguments

base.cov variance/covariance matrix
...

Provides coefficient of determination for linear models, using variance/correlation matrix.
**Value**

A list with several fit indices

- r.squared.xy $R^2_{XY}$
- p.squared.yx $P^2_{YX}$

See `mlmWithCov`

**References**


**See Also**

Other fit indices: `da.betareg.fit()`, `da.dynlm.fit()`, `da.glm.fit()`, `da.lm.fit()`, `da.lmWithCov.fit()`, `da.lmerMod.fit()`

---

**dominanceAnalysis**

*Dominance analysis for OLS (univariate and multivariate), GLM and LMM models*

**Description**

Dominance analysis for OLS (univariate and multivariate), GLM and LMM models

**Usage**

```r
dominanceAnalysis(
  x,
  constants = c(),
  terms = NULL,
  fit.functions = "default",
  data = NULL,
  null.model = NULL,
  link.betareg = NULL,
  ...
)
```

**Arguments**

- `x` : lm, glm, lmer model
- `constants` : vector of predictors to remain unchanged between models
- `terms` : vector of terms to be analyzed. By default, obtained from the model
- `fit.functions` : Name of the method used to provide fit indices
data: optional data.frame
null_model: for mixed models, null model against to test the submodels
link.betareg: for betareg, link function to use.
...: Other arguments provided to lm or lmer (not implemented yet)

Value

- predictors: Vector of predictors.
- constants: Vector of constant variables.
- terms: Vector of terms to be analyzed.
- fit.functions: Vector of fit indices names.
- fits: List with raw fits indices. See daRawResults.
- contribution.by.level: List of mean contribution of each predictor by level for each fit index. Each element is a data.frame, with levels as rows and predictors as columns, for each fit index.
- contribution.average: List with mean contribution of each predictor for all levels. These values are obtained for every fit index considered in the analysis. Each element is a vector of mean contributions for a given fit index.
- complete: Matrix for complete dominance.
- conditional: Matrix for conditional dominance.
- general: Matrix for general dominance.

Definition of Dominance Analysis

Budescu (1993) developed a clear and intuitive definition of importance in regression models, that states that a predictor’s importance reflects its contribution in the prediction of the criterion and that one predictor is ‘more important than another’ if it contributes more to the prediction of the criterion than does its competitor at a given level of analysis.

Types of dominance

The original paper (Bodescu, 1993) defines that variable \( X_1 \) dominates \( X_2 \) when \( X_1 \) is chosen over \( X_2 \) in all possible subset of models where only one of these two predictors is to be entered. Later, Azen & Bodescu (2003), name the previously definition as ‘complete dominance’ and two other types of dominance: conditional and general dominance. Conditional dominance is calculated as the average of the additional contributions to all subset of models of a given model size. General dominance is calculated as the mean of average contribution on each level.

Fit indices availables

To obtain the fit-indices for each model, a function called da.<model>.fit is executed. For example, for a lm model, function da.lm.fit provides \( R^2 \) values. Currently, seven models are implemented:

- lm: Provides \( R^2 \) or coefficient of determination. See da.lm.fit
dominanceAnalysis


lmerMod Provides four fit indices recommended by Lou & Azen (2012). See da.lmerMod.fit

lmWithCov Provides $R^2$ for a correlation/covariance matrix. See lmWithCov to create the model and da.lmWithCov.fit for the fit index function.

mlmWithCov Provides both $R^2_{XY}$ and $P^2_{XY}$ for multivariate regression models using a correlation/covariance matrix. See mlmWithCov to create the model and da.mlmWithCov.fit for the fit index function.

dynlm Provides $R^2$ for dynamic linear models. There is no literature reference about using dominance analysis on dynamic linear models, so you’re warned!. See da.dynlm.fit.

betareg Provides pseudo-$R^2$, Cox and Snell(1989), McFadden (1974), and Estrella (1998). You could set the link function using link.betareg if automatic detection of link function doesn’t work.

See da.betareg.fit

References


Examples

data(longley)
lm.1<-lm(Employed~.,longley)
da<-dominanceAnalysis(lm.1)
print(da)
summary(da)
plot(da,which.graph='complete')
plot(da,which.graph='conditional')
plot(da,which.graph='general')

# Maintaining year as a constant on all submodels
da.no.year<-dominanceAnalysis(lm.1,constants='Year')
print(da.no.year)
summary(da.no.year)
plot(da.no.year, which.graph='complete')

# Parameter terms could be used to group variables
da.terms=c(GNP.rel='GNP.deflator+GNP',
           pop.rel='Unemployed+Armed.Forces+Population+Unemployed',
           year='Year')
da.grouped<-dominanceAnalysis(lm.1, terms=da.terms)
print(da.grouped)
summary(da.grouped)
plot(da.grouped, which.graph='complete')

---

dominanceBriefing  Retrieve a briefing for complete, conditional and general dominance

Description
Retrieve a briefing for complete, conditional and general dominance

Usage
dominanceBriefing(da.object, fit.functions = NULL, abbrev = FALSE)

Arguments

da.object  a dominanceAnalysis object
fit.functions name of the fit indices to retrieve. If NULL, all fit indices will be retrieved
abbrev      if TRUE

Value
a list. Each element is a data.frame, that comprises the dominance analysis for a specific fit index. Each data.frame have the predictors as row and each column reports the predictors that are dominated for each predictor

See Also
Other retrieval methods: averageContribution(), contributionByLevel(), dominanceMatrix(), getFits() 

Examples

# For matrix or data.frame
data(longley)
da.longley<-dominanceAnalysis(lm(Employed~., longley))
dominanceBriefing(da.longley, abbrev=FALSE)
dominanceBriefing(da.longley, abbrev=TRUE)
dominanceMatrix

Retrieve or calculates a dominance matrix for a given object

Description
This method calculates or retrieves a dominance matrix.
This method allows a common interface to retrieve all dominance matrices from dominanceAnalysis objects.

Usage

dominanceMatrix(x, ...)

## S3 method for class 'data.frame'
dominanceMatrix(x, undefined.value = 0.5, ordered = FALSE, ...)

## S3 method for class 'matrix'
dominanceMatrix(x, undefined.value = 0.5, ordered = FALSE, ...)

## S3 method for class 'dominanceAnalysis'
dominanceMatrix(
  x,
  type,
  fit.functions = NULL,
  drop = TRUE,
  ordered = FALSE,
  ...
)

Arguments

x       matrix (calculate) or dominanceAnalysis (retrieve)
...
undefined.value
value when no dominance can be established
ordered Logical. If TRUE, sort the output according to dominance.
type     type of dominance matrix to retrieve. Could be complete, conditional or general
fit.functions
name of the fit indices to retrieve. If NULL, all fit indices will be retrieved
if TRUE and just one fit index is available, returns a matrix. Else, returns a list

Details
To calculate a dominance matrix from a matrix or dataframe, use
dominanceMatrix(x,undefined.value).
To retrieve the dominance matrices from a dominanceAnalysis object, use
dominanceMatrix(x,type,fit.function,drop)
getFits

Value

for matrix and data-frame, returns a matrix representing dominance. 1 represents domination of the row variable over the column variable, 0 dominance of the column over the row variable. Undefined dominance is represented by undefined.value parameter. For dominanceAnalysis object, returns a matrix, if drop parameter if TRUE and just one index is available. Else, a list is returned, with keys as name of fit-indices and values as matrices, as described previously.

See Also

Other retrieval methods: averageContribution(), contributionByLevel(), dominanceBriefing(), getFits()

Examples

# For matrix or data.frame
mm<-data.frame(a=c(5,3,2),b=c(4,2,1),c=c(5,4,3))
dominanceMatrix(mm)
# For dominanceAnalysis
data(longley)
da.longley<-dominanceAnalysis(lm(Employed~.,longley))
dominanceMatrix(da.longley,type="complete")

getFits

Retrieve fit matrix or matrices

description

Retrieve fit matrix or matrices for a given dominanceAnalysis object

Usage

getFits(da.object, fit.functions = NULL)

Arguments

da.object dominanceAnalysis object
fit.functions name of the fit indices to retrieve. If NULL, all fit indices will be retrieved

Value

a list. Key corresponds to fit-index and the value is a matrix, with fits values

See Also

Other retrieval methods: averageContribution(), contributionByLevel(), dominanceBriefing(), dominanceMatrix()
**lmR2**

*Calculates several measures of fit for Linear Mixed Models based on Lou and Azen (2013) text. Models could be lmer or lme models*

**Description**

Calculates several measures of fit for Linear Mixed Models based on Lou and Azen (2013) text. Models could be lmer or lme models

**Usage**

```r
lmR2(m.null, m.full)
```

**Arguments**

- `m.null`: Null model (only with random intercept effects)
- `m.full`: Full model

**Value**

lmR2 class

---

**lmWithCov**

*Uses covariance/correlation matrix for calculate OLS*

**Description**

Calculate regression coefficients and $R^2$ for an OLS regression. Could be used with dominanceAnalysis to perform a dominance analysis without the original data.

**Usage**

```r
lmWithCov(f, x)
```

**Arguments**

- `f`: formula for lm model
- `x`: correlation/covariance matrix
Value

- **coef**: regression coefficients
- **r.squared**: $R^2$ or coefficient of determination
- **formula**: formula provided as parameter
- **cov**: covariance/correlation matrix provided as parameter

Examples

```r
cov.m<--matrix(c(1,0.2,0.3, 0.2,1,0.5,0.3,0.5,1),3,3,
dimnames=list(c("x1","x2","y"),c("x1","x2","y")))
lm.cov<-lmWithCov(y~x1+x2,cov.m)
da<-dominanceAnalysis(lm.cov)
```

---

**mlmWithCov**

*Uses covariance/correlation matrix to calculate multivariate index of fit*

Description

Calculate $R^2_{XY}$ and $P^2_{YX}$ for multivariate regression. Could be used with `dominanceAnalysis` to perform a multivariate dominance analysis without original data.

Usage

```r
mlmWithCov(f, x)
```

Arguments

- **f**: formula. Should use `cbind(y1,y2,...,yk)~x1+x2+..+xp`
- **x**: correlation/covariance matrix

Value

- **r.squared.xy**: $R^2_{XY}$ of the regression
- **p.squared.yx**: $P^2_{YX}$ of the regression
- **formula**: formula provided as parameter
- **cov**: covariance/correlation matrix provided as parameter
Examples

```r
library(car)
cor.m<-matrix(c(
  1.0000000, 0.7951377, 0.2617168, 0.6720053, 0.3390278,
  0.7951377, 1.0000000, 0.3341037, 0.5876337, 0.3404206,
  0.2617168, 0.3341037, 1.0000000, 0.3703162, 0.2114153,
  0.6720053, 0.5876337, 0.3703162, 1.0000000, 0.3548077,
  0.3390278, 0.3404206, 0.2114153, 0.3548077, 1.0000000),
5,5,
byrow = TRUE,
dimnames = list(
  c("na","ss","SAT","PPVT","Raven"),
  c("na","ss","SAT","PPVT","Raven")))
lwith<-mlmWithCov(cbind(na,ss)~SAT+PPVT+Raven,cor.m)
da<-dominanceAnalysis(lwith)
print(da)
summary(da)
```

---

**plot.dominanceAnalysis**

*Plot for a dominanceAnalysis object*

**Description**

Plot for a `dominanceAnalysis` object

**Usage**

```r
## S3 method for class 'dominanceAnalysis'
plot(
  x,
  which.graph = c("general", "complete", "complete_no_facet", "conditional"),
  fit.function = NULL,
  ...
)
```

**Arguments**

- `x` a `dominanceAnalysis` object
- `which.graph` which graph to plot
- `fit.function` name of the fit indices to retrieve. If NULL, first index will be used
- `...` unused

**Value**

a ggplot object
Examples

data(longley)
lm.1<-lm(Employed~.,longley)
da<-dominanceAnalysis(lm.1)
# By default, plot complete dominance of first fit function
plot(da)
# Parameter which.graph defines dominance to plot
plot(da,which.graph='conditional')
plot(da,which.graph='general')

replaceTermsInString  Replace terms by name using the terms definition

Description

Replace terms by name using the terms definition

Usage

replaceTermsInString(string, replacement)

Arguments

string        string to be updated
replacement   string with replacement for strings. values are replaced by names

tropicbird  Distribution of a tropical native bird species inhabiting a small oceanic island.

Description

The dataset contains information about points distributed across a small oceanic island (Soares, 2017). In each of these points, a 10-minute count was carried out to record the species presence (assuming 1 if the species was present, or 0 if it was absent). The species’ presence/absence is the binary response variable (i.e., dependent variable). Additionally, all sampled points were characterized by multiple environmental variables.

Usage

tropicbird
Format

A data frame with 2398 rows and 8 variables:

ID  Point identification
rem remoteness is an index that represents the difficulty of movement through the landscape, with the highest values corresponding to the most remote areas
land land use is an index that represents the land-use intensification, with the highest values corresponding to the more humanized areas (e.g., cities, agricultural areas, horticultures, oil-palm monocultures)
alp altitude is a continuous variable, with the highest values corresponding to the higher altitude areas
slo slop is a continuous variable, with the highest values corresponding to the steepest areas
rain rainfall is a continuous variable, with the highest values corresponding to the rainy wet areas
cost coast distance to the coast is the minimum linear distance between each point and the coast line, with the highest values corresponding to the points further away from the coastline
pres Species presence

Source


using-fit-indices Provides fit indices for different regression models.

Description

dominanceAnalysis tries to infer, based on the class of the model provided, the appropriate fit indices, using the scheme da.CLASS.fit for name. This method has two interfaces, one for retrieving the names of the fit indices, and another to retrieve the indices based on the data.

Arguments

data Complete data set containing the variables in the model.
null.model Null model only needed for HLM models.
base.cov Required if only a covariance/correlation matrix is provided.
family.glm family param for glm models.

Details

Interfaces are:

- da.CLASS.fit("names") returns a vector with names for fit indices
- da.CLASS.fit(data, null.model, base.cov=NULL, family.glm=NULL) returns a function with one parameter, the formula to calculate the submodel.
Index

∗Topic datasets
   tropicbird, 22

averageContribution, 4, 7, 16, 18

bootAverageDominanceAnalysis, 5
bootDominanceAnalysis, 2, 3, 6

dominanceAnalysis, 2, 3, 6, 7, 13, 19–21, 23
dominanceAnalysis-package, 2
domianceBriefing, 5, 7, 16, 18
dominanceMatrix, 5, 7, 16, 17, 18

gets, 5, 7, 16, 18, 18

lmmR2, 19
lmWithCov, 2, 12, 15, 19

mlmWithCov, 2, 13, 15, 20

plot.dominanceAnalysis, 21
replaceTermsInString, 22
tropicbird, 22

using-fit-indices, 8–11, 23