Package 'lmdme'

March 26, 2013

Type Package

Title Linear Model decomposition for Designed Multivariate Experiments

Version 1.0.0

Date 2012-09-03

Author Cristobal Fresno and Elmer A. Fernandez

Maintainer Cristobal Fresno <cristobalfresno@gmail.com>

Description linear ANOVA decomposition of Multivariate Designed Experiments implementation based on limma lmFit. Features: i) Flexible formula type interface, ii) Fast limma based implementation, iii) p and F values over deflacted coefficients and iv) ploting functions for PCA and PLS

License GPL (>=2)

biocViews Microarray, OneChannel, TwoChannel, Bioinformatics,Visualization, AssayDomains, DifferentialExpression,ExperimentData, Cancer

URL http://200.45.112.41/bdmg/?page_id=38

Imports stats

Depends R (>= 2.14.1), methods, limma, pls

Suggests stemHypoxia

Enhances parallel

Collate 'lmDME-Class.R' 'lmDME-lmdme.R' 'lmDME-getters.R''lmDME-printshow.R' 'lmDME-padjust.R' 'lmDME-leverage.R''lmDME-permutation.R' 'lmDME-decomposition.R' 'lmDME-biplot.R''lmDME-screeplot.R' 'lmDME-loadingplot.R'

R topics documented:

---

biplot *Plot a* biplot *of a lmDME object*

Description

Plot a biplot over each decomposed "pca" or "plsr" present in components lmDME object's slot.

Usage

```
  ## S4 method for signature 'lmDME'
biplot(x, comp=1:2, xlab = NULL, ylab =
   NULL, term=NULL, mfcol, ...)
```

Arguments

x	lmDME class object.
comp	a tow component vector with the PC components to plot. Default comp=1:2,
xlab	character for the x-label title for PCA biplots
ylab	character for the y-label title for PCA biplots
term	character with the corresponding term/s for biploting. Default value is NULL in order to obtain every available biplot/s.
mfcol	numeric vector for par layout. If missing mfcol=c(1,2) will be used if more than one biplot is available. Use mfcol==NULL to override par call inside biplot function.
...	additional parameters for biplot.prcomp (pca) or biplot.mrv (plsr)

Value

ploted biplot/s of the components slot of the given lmDME object. If par() is called before this function, the biplots can be arranged in the same window

Author(s)

Cristobal Fresno and Elmer A Fernandez

See Also

prcomp, plsr, biplot.princomp, biplot.mvr

Examples

```
if (require(stemHypoxia)) {
  data(stemHypoxia)
  ##Just to make a balance dataset in the Fisher sense (2 samples per time*oxygen levels)
  design<-design[design$time %in% c("0.5","1","5") & design$oxygen %in% c("1","5","21"),]
  design$time <-as.factor(design$time)
  design$oxygen<-as.factor(design$oxygen)
  rownames(M)<-M[,1]
  M <- M[,colnames(M) %in% design$samplename] #Keeping appropiate samples only
  ##ANOVA decomposition
  fit <- lmdme(model=~time+oxygen+time:oxygen,data=M,design=design)
 ##ASCA for all the available terms, over those subjects/genes where at least one interaction coefficient is statistically di
  id<-F.p.values(fit,term="time:oxygen")[[1]]<0.001
  decomposition(fit,decomposition="pca",scale="row",subset=id)
  ## Not run:
  par(mfrow=c(2,2)); biplot(fit,xlabs=rep("o",sum(id)),mfcol=NULL) ##Do not call par inside
  biplot(fit,xlabs=rep("o",sum(id)),term="time")##Just the term of interest
 biplot(fit,xlabs=rep("o",sum(id)),term=c("time","oxygen"),mfcol=c(1,1))##In separate graphics
  biplot(fit,xlabs=rep("o",sum(id)),mfcol=c(1,3))##All term in the same graphic

## End(Not run)
  ##Now using plsr over interaction coefficients
  decomposition(fit,decomposition = "plsr", term ="time:oxygen",scale="row", subset=id)
}
## Not run:
par(mfrow=c(2,2))
biplot(fit, which = "x", mfcol=NULL) ##plsr biplot by default which="x"
biplot(fit, which = "y", mfcol=NULL) ##Other alternatives to which
biplot(fit, which = "scores", mfcol=NULL)
biplot(fit, which = "loadings", mfcol=NULL, xlabs=rep("o",sum(id)))

## End(Not run)
```

---

decomposition decomposition *of lmDME object*

Description

This function calculates the decomposition of variance or covariance structure using Principal Components Analysis (PCA) or Partial Least Squared Regression (PLSR), over ANOVA decomposed lmDME object. In this context, in a two factor experimental design with interaction, the linear model of the i-th observation (gene) can be written:
$X = \mu + XA + XB + XAB + \epsilon$
where

- X stands for the observed value
- the intercept the intercept μ
- XA, XB and XAB are the first, second and interaction coefficients respectively
- The error term ϵ $N(0, \sigma^2)$.

The the model is iterative decomposed in a step by step fashion decomposing one term at each time by calling lmdme constructor:

1. Step 1: $X = \mu + E1$
2. Step 2: E1=XA+E2
3. Step 3: E2=XB+E3
4. Step 4: E3=XAB+E4.

Then, if we apply PCA over the i-th step using Ei-1 matrix it is known as *APCA* but if applied over the coefficients Xi it is called *ASCA*. The same decomposition schema can also be used with PLSR.

Usage

```
## S4 method for signature 'lmDME'
decomposition(object,
  decomposition = c("pca", "plsr"), term = NULL,
  subset = 1:nrow(object@residuals[[1]]),
  type = c("coefficient", "residual"),
  scale = c("none", "row", "column"), Omatrix, ...)
```

Arguments

object	lmDME class object.
decomposition	character to indicate the decomposition to be carry out, i.e., *PCA* or *PLSR*. Default value is PCA.
term	character specifying the model term to perform the decomposition (e.g. "time" or "time:concentration" for interaction term). If term is not specified (i.e. missing) it performs the analysis over all the model terms.
subset	subset of indididuals (rows) to be included in the analysis. By default all the indididuals are included.
type	character to indicate over which regressor matrix perform the decomposition ("coefficient" or "residual"). The intercept term is not included in the results, as it can be directly analyzed with the original M data.frame. Default value is "coefficient" a.k.a. ASCA. Note that "residual" performs PCA or PLS over the i-th residual Ei-1=Xi+Ei and not the residuals of the i-th model Ei.
scale	character "row", "column" or "none" to indicate if the matrix should be scaled by the row, column or not respectively. Default value is "none".
Omatrix	the output matrix for PLSR only. If the parameter is missing, the output matrix will be an identity matrix for the ASCA, otherwhise the design matrix corresponding to the specified term for APCA.
...	additional parameters for prcomp or plsr functions, according to decomposition call.

Value

Internal update of the "components" slot of the lmDME object, which is a list of prcomp or a list of mvr (plsr) objects using the given term parameter. If missing(term), the length of the list equal the number of decomposed models minus the Intercept term for coefficients or the length of decomposed models for residual decomposition.

Author(s)

Cristobal Fresno and Elmer A Fernandez

References

1. Smilde AK, Jansen JJ, Hoefsloot HCJ, Lamer RAN, Van der Greef J, Timmerman ME (2005) ANOVA-simultaneaus component analysis (ASCA): a new tool for analyzing designed metabolomics data, Bioinformatics 21,13,3043 DOI:/10.1093/bioinformatics/bti476
2. Zwanenburg G, Hoefsloot HCJ, Westerhuis JA, Jansen JJ, Smilde AK (2011) ANOVA.Principal component analysis and ANOVA-simultaneaus component analysis: a comparison J. Chemometrics 25:561-567 DOI:10.1002/cem.1400

See Also

prcomp, plsr

Examples

```
if (require(stemHypoxia)) {
  data(stemHypoxia)
  ##Just to make a balance dataset in the Fisher sense (2 samples per time*oxygen levels)
  design<-design[design$time %in% c("0.5","1","5") & design$oxygen %in% c("1","5","21"),]
  design$time <-as.factor(design$time)
  design$oxygen<-as.factor(design$oxygen)
  rownames(M)<-M[,1]
  M <- M[,colnames(M) %in% design$samplename] #Keeping appropiate samples only
  ##ANOVA decomposition
  fit <- lmdme(model=~time+oxygen+time:oxygen,data=M,design=design)
 ##Just a copy of the same fit object and to perform analysis over those subjects/genes where at least one interaction coe
  asca <- fit; apca <-fit; plsr.residuals <- fit; plsr.coefficients <-fit
  id<-F.p.values(fit,term="time:oxygen")[[1]]<0.001
  ##ASCA and APCA decomposition for every available term.
  decomposition(asca,decomposition="pca", subset=id, scale="row")
  decomposition(apca,decomposition="pca", subset=id, scale="row", type="residual")
  asca <- components(asca) ##get the coefficients prcomp objects
  apca <- components(apca) ##get the residuals prcomp objects
  ##PLSR decomposition for residuals and coefficients
  decomposition(plsr.coefficients,decomposition="plsr", subset=id, scale="row")
  decomposition(plsr.residuals,decomposition="plsr", subset=id, scale="row", type="residual")
  plsr.coefficients <- components(plsr.coefficients) ##get the coefficients plsr objects
  plsr.residuals <- components(plsr.residuals) ##get the residuals plsr objects
}
```

---

fitted.values *Getters for lmDME object*

Description

Obtain lmDME slot information, acording to the given function call (see Values). If term parameter is not specified, it will return all the available terms, otherwise just the one specified.

Usage

```
  ## S4 method for signature 'lmDME'
fitted.values(object,term)
```

```
  ## S4 method for signature 'lmDME'
fitted(object,term)

  ## S4 method for signature 'lmDME'
coef(object,term)

  ## S4 method for signature 'lmDME'
coefficients(object,term)

  ## S4 method for signature 'lmDME'
resid(object,term)

  ## S4 method for signature 'lmDME'
residuals(object,term)

  F.p.values(object, term = NULL)

  ## S4 method for signature 'lmDME'
F.p.values(object, term = NULL)

  p.values(object, term = NULL)

  ## S4 method for signature 'lmDME'
p.values(object, term = NULL)

  modelDecomposition(object, term = NULL)

  ## S4 method for signature 'lmDME'
modelDecomposition(object, term = NULL)

  components(object, term = NULL)

  ## S4 method for signature 'lmDME'
components(object, term = NULL)

  componentsType(object, term = NULL)

  ## S4 method for signature 'lmDME'
componentsType(object, term = NULL)

  model(object)

  ## S4 method for signature 'lmDME'
model(object)

  design(object)

  ## S4 method for signature 'lmDME'
design(object)
```

Arguments

object	lmDME class object.

term — character with the corresponding term/s to return. Default value is NULL in order to return every available term/s.

Value

according to the call one of the following objects can be returned

design — used experiment design data.frame.

model — used decompose formula.

modelDecomposition — list of decomposed model formulas.

residuals or resid, coef or coefficients, fitted or fitted.values, p.values or F.p.values — list of appropieate slot where each item is a matrix that will have G rows(individuals) x k columns(levels of the corresponding model term).

components — list with corresponding PCA or PLSR term according to decomposition function call.

componentsType — character name vector with the information of the components calculation.

Author(s)

Cristobal Fresno and Elmer A Fernandez

See Also

lmdme, decomposition, print, show

Examples

```
if (require(stemHypoxia)) {
  data(stemHypoxia)
  ##Just to make a balance dataset in the Fisher sense (2 samples per time*oxygen levels)
  design<-design[design$time %in% c("0.5","1","5") & design$oxygen %in% c("1","5","21"),]
  design$time <-as.factor(design$time)
  design$oxygen<-as.factor(design$oxygen)
  rownames(M)<-M[,1]
  M <- M[,colnames(M) %in% design$samplename] #Keeping appropiate samples only
  ##ANOVA decomposition
  fit <- lmdme(model=~time+oxygen+time:oxygen,data=M,design=design)
  fit.model  <- model(fit) #The model formula used
  fit.design <- design(fit)#The design data.frame used
  fit.modelDecomposition<- modelDecomposition(fit) #How the decomposition was carried out
  ##Getting the coefficients
  timeCoef <- coef(fit,term="time")  #or coef(fit) to get all term coefficients
  fit.p.values <- p.values(fit,term="time") #for the same term
  fit.f.values <- F.p.values(fit,term="time") #or the F-values
  ##Getting the residuals or fitted values
  interactionResid <- resid(fit, term="time:oxygen") #or resid(fit) to get all term residuals
  oxygenDFit <- fitted(fit, term="(Intercept)") #or fitted(fit) to get all term fitted values
}
```

leverage	leverage *test of Linear model for Designed Multivariate Experiments objects*

Description

This function calculates the leverage test for each individual using the Principal Component Analysis (comps function) over the coefficients of the given decomposed model term.

Usage

```
  ## S4 method for signature 'lmDME'
leverage(object, comps = 1:2, term,
   level = 0.95)
```

Arguments

object	lmDME class object.
comps	a numeric vector indicating the PCA component indexes to keep. Default the first two components (1:2)
term	a character specifying the model term.
level	the quantile level. Default value 0.95

Value

data.frame with the following fields

leverage	numeric for the corresponding row leverage
over	logical indicating if the leverage > quantile(leverage,level) for the given decomposed term

Author(s)

Cristobal Fresno and Elmer A Fernandez

References

Tarazona S, Prado-Lopez S, Dopazo J, Ferrer A, Conesa A, Variable Selection for Multifactorial Genomic Data, Chemometrics and Intelligent Laboratory Systems, 110:113-122 (2012)

See Also

prcomp, quantile

Examples

```
if (require(stemHypoxia)) {
  data(stemHypoxia)
  ##Just to make a balance dataset in the Fisher sense (2 samples per time*oxygen levels)
  design<-design[design$time %in% c("0.5","1","5") & design$oxygen %in% c("1","5","21"),]
  design$time <-as.factor(design$time)
  design$oxygen<-as.factor(design$oxygen)
  rownames(M)<-M[,1]
  M <- M[,colnames(M) %in% design$samplename] #Keeping appropiate samples only
  ##ANOVA decomposition
  fit <- lmdme(model=~time+oxygen+time:oxygen,data=M,design=design)
  ##Leverages for the first two Principal Components and q95
  leverages <- leverage(fit,term="time:oxygen")
  ##Leverages for the first three Principal Components and q99
  leverages <- leverage(fit,comps=1:3,term="time:oxygen",level=0.99)
}
```

---

lmdme *High level constructor of lmDME class object*

Description

Linear model ANOVA decomposition of Designed Multivariate Experiments based on limma lmFit implementation. For example in a two factor experimental design with interaction, the linear model of the i-th observation (gene) can be written:
$X = \mu + A + B + AB + \epsilon$
where

- X stands for the observed value
- the intercept the intercept μ
- A, B and AB are the first, second and interaction terms respectively
- The error term ϵ $N(0, \sigma^2)$.

The the model is iterative decomposed in a step by step fashion decomposing one term at each time:

1. The intercept is estimated using $X = \mu + E1$
2. The first factor (A) using E1=A+E2
3. The second factor (B) using E2=B+E3
4. The interaction (AB) using E3=AB+E4.

For each decomposed step the model, residuals, coefficients, p-values and F-values are stored in a list container, so their corresponding length is equal to the number of model terms + 1 (the intercept).

Usage

```
  ## S4 method for signature 'formula,ANY,data.frame'
lmdme(model, data,
    design, Bayes = FALSE, verbose = FALSE, ...)
```

Arguments

model	formula object to carry out the decomposition
data	matrix or data.frame with individuals/genes (rows) and samples/conditions (columns)
design	data.frame with the design of the experiment, (rows) samples/conditions as in data columns and as many columns to indicate the factors presents in each sample.
Bayes	Should limma estimate empirical Bayes statistics, i. e., moderated t-statistics? Default value is FALSE.
verbose	Should the process progress be printed? Default value is FALSE.
...	Additional parameters for lmFit function

Value

lmDME	lmDME class object with the corresponding completed slots according to the given model

Note

use **lmdme** high level constructor for the creation of the class instead of directely calling its constructor by means of new.

Author(s)

Cristobal Fresno and Elmer A Fernandez

References

1. Smyth, G. K. (2005). Limma: linear models for microarray data. In: Bioinformatics and Computational Biology Solutions using R and Bioconductor. R. Gentleman, V. Carey, S. Dudoit, R. Irizarry, W. Huber (eds), Springer, New York, pages 397–420.

See Also

decomposition, lmFit

Examples

```
if (require(stemHypoxia)) {
  data(stemHypoxia)
  ##Just to make a balance dataset in the Fisher sense (2 samples per time*oxygen levels)
  design<-design[design$time %in% c("0.5","1","5") & design$oxygen %in% c("1","5","21"),]
  design$time <-as.factor(design$time)
  design$oxygen<-as.factor(design$oxygen)
  rownames(M)<-M[,1]
  M <- M[,colnames(M) %in% design$samplename] #Keeping appropiate samples only
  ##ANOVA decomposition
  fit <- lmdme(model=~time+oxygen+time:oxygen,data=M,design=design)
}
```

lmDME-class *lmDME S4 class: Linear model decomposition for Designed Multivariate Experiments.*

Description

Linear Model ANOVA decomposition of Designed Multivariate Experiments based on limma lmFit implementation. For example in a two factor experimental design with interaction, the linear model of the i-th observation (gene) can be written:
$X = \mu + A + B + AB + \epsilon$
where

- X stands for the observed value
- the intercept the intercept μ
- A, B and AB are the first, second and interaction terms respectively
- The error term ϵ $N(0, \sigma^2)$.

The the model is iterative decomposed in a step by step fashion decomposing one term at each time:

1. The intercept is estimated using $X = \mu + E1$
2. The first factor (A) using E1=A+E2
3. The second factor (B) using E2=B+E3
4. The interaction (AB) using E3=AB+E4.

For each decomposed step the model, residuals, coefficients, p-values and F-values are stored in a list container, so their corresponding length is equal to the number of model terms + 1 (the intercept).

Features

1. Flexible formula type interface,
2. Fast limma based implementation based on lmFit,
3. p and F-values over deflacted coefficients,
4. ploting functions for PCA and PLS.

Slots

- design: data.frame with experimental design.
- model: formula with the designed model to be decomposed.
- modelDecomposition: list with the model formula obtained for each deflation step.
- residuals: list of residual matrices G rows(genes) x N columns(arrays-designed meassurements).
- coefficients: list of coefficient matrices. Each matrix will have G rows(genes) x k columns(levels of the corresponding model term).
- p.values: list of p-value matrices.
- F.p.values: list with corresponding F-p-values vectors for each individual.
- components: list with corresponding PCA or PLS components for the selected term/s.
- componentsType: name character vector to keep process trace of the variance/covariance components slot: decomposition ("pca" or "pls"), type ("apca" for ANOVA-PCA or "asca" for ANOVA-SCA) and scale ("none", "row" or "column")

lmDME-general-functions

print, show Basic output for lmDME class

summary Basic statistics for lmDME class

design, model, modelDecomposition, residuals, coefficients, p.values, F.p.values, components and componentsTyp Getters for their respective slots.

ANOVA-linear-decomposition-functions

lmdme Function that produce the complete ANOVA decomposition based on model specification through formula interface. Technicaly is a high level wrapper of initialize function.

modelDecomposition Getter for decomposed used formula in each step

p.adjust Adjust coefficients p-values for Multiple Comparisons Test.

Fpvalues, pvalues Getters for corresponding model decomposed asociated coefficient statistics in each step, for each observation

residuals, resid, coef, coefficients, fitted.values, fitted Getters for corresponding model decomposed in each step

permutation Produces de specified lmDME in addition to the requied permuted objects (sampling the columns of data), usign the same parameters to fit the model.

variance-covariance-decomposition-functions

decomposition Function to perform PCA or PLS over the ANOVA deflacted terms. PCA can be performed over E1, E2 or E3 and it is refered as ANOVA-PCA (APCA) but, if it is performed over the coefficients it is refered as ANOVA-SCA (ASCA). On the other hand PLSR is based on pls library and if it is performed on coefficients (ASCA like) it uses the identity matrix for output co-variance maximization or can be carried out over the E1, E2 or E3 (APCA like) using the design matrix as output.

components Getter for PCA or PLS decomposed models.

componentsType Getter for componentsType slot.

leverage Leverage calculation over PCA (APCA or ASCA) terms.

biplot Biplots for PCA or PLSR decomposed terms.

screeplot Screeplot over each decomposed PCA decomposed term.

loadingplot Loafingplot for PCA interaction terms.

Author(s)

Cristobal Fresno and Elmer A Fernandez

References

1. Smilde AK, Jansen JJ, Hoefsloot HCJ, Lamer RAN, Van der Greef J, Timmerman ME (2005) ANOVA-simultaneaus component analysis (ASCA): a new tool for analyzing designed metabolomics data, Bioinformatics 21,13,3043 DOI:/10.1093/bioinformatics/bti476
2. Zwanenburg G, Hoefsloot HCJ, Westerhuis JA, Jansen JJ, Smilde AK (2011) ANOVA.Principal component analysis and ANOVA-simultaneaus component analysis: a comparison J. Chemometrivcs 25:561-567 DOI:10.1002/cem.1400
3. Tarazona S, Prado-Lopez S, Dopazo J, Ferrer A, Conesa A (2012) Variable Selection for Multifactorial Genomic Data, Chemometrics and Intelligent Laboratory Systems, 110:113-122

See Also

lmdme, decomposition, biplot, loadingplot and additional related lmDME class functions.

---

loadingplot loadingplot *of interaction PCA decomposed lmDME object*

Description

This function plots the PCA loadings for a given interaction (A:B) lmDME object's components slot, for the given "pc" component. The user can choose which term (A or B) is used for x-axis and y-axis functions (B or A) respectively.

Usage

```
  ## S4 method for signature 'lmDME'
loadingplot(object, term.x, term.y,
   pc=1, ord.x, col, ...)
```

Arguments

object	lmDME class object.
term.x,term.y	character indicating the model principal factor for the interaction term (term.x:term.y or term.y:term.x) for the corresponding x or y axis.
pc	integer indicating which principal component loading is to be plotted on the y-axis. Default value is 1.
col	which color to use for each level present in term.y.
ord.x	numeric indicating the term.x levels order, for plotting purpoouses. If missing the levels order is used.
...	additional parameters for maplot.

Value

loading plot of the selected interaction (term.x:term.y) components lmDME object's slot, if PCA decomposition was applied.

Author(s)

Cristobal Fresno and Elmer A Fernandez

Examples

```
if (require(stemHypoxia)) {
  data(stemHypoxia)
  ##Just to make a balance dataset in the Fisher sense (2 samples per time*oxygen levels)
  design<-design[design$time %in% c("0.5","1","5") & design$oxygen %in% c("1","5","21"),]
  design$time <-as.factor(design$time)
  design$oxygen<-as.factor(design$oxygen)
  rownames(M)<-M[,1]
  M <- M[,colnames(M) %in% design$samplename] #Keeping appropiate samples only
  ##ANOVA decomposition
  fit <- lmdme(model=~time+oxygen+time:oxygen,data=M,design=design)
```

```
 ##ASCA for all the available terms, over those subjects/genes where at least one interaction coefficient is statistically di
 id<-F.p.values(fit,term="time:oxygen")[[1]]<0.001
 decomposition(fit,decomposition="pca",scale="row", subset=id)
}
## Not run:
 loadingplot(fit, term.x="time", term.y="oxygen")
 loadingplot(fit, term.x="oxygen", term.y="time") ##Or change the axis order
 loadingplot(fit, term.x="time", term.y="oxygen",pc=2) ##Or change the PC to display
 loadingplot(fit, term.x="time", term.y="oxygen",ord.x=3:1) ##Or the order of x-levels

## End(Not run)
```

---

p.adjust p.adjust *of p-values for Multiple Test Comparisons Correction*

Description

Given a set of p-values, returns p-values adjusted using one of several methods.

Usage

```
  ## S4 method for signature 'ANY'
p.adjust(p, ...)

  ## S4 method for signature 'lmDME'
p.adjust(p,term,method)
```

Arguments

p	numeric vector of p-values as in stats::p.adjust or lmDME class object.
method	correction method availables in p.adjust.methods.
term	character with the corresponding term to return.
...	other arguments.

Value

according to the call one of the following objects can be returned

numeric	vector of adjusted p-values
matrix	for lmDME object If term!=NULL, the corresponding character is looked up within list of p.values returning the associated matrix of G rows(individuals) x k columns(levels of the corresponding model term) with the adjusted p-values.

Author(s)

Cristobal Fresno and Elmer A Fernandez

See Also

p.adjust, p.adjust.methods

Examples

```
if (require(stemHypoxia)) {
  data(stemHypoxia)
  ##Just to make a balance dataset in the Fisher sense (2 samples per time*oxygen levels)
  design<-design[design$time %in% c("0.5","1","5") & design$oxygen %in% c("1","5","21"),]
  design$time <-as.factor(design$time)
  design$oxygen<-as.factor(design$oxygen)
  rownames(M)<-M[,1]
  M <- M[,colnames(M) %in% design$samplename] #Keeping appropiate samples only
  ##ANOVA decomposition
  fit <- lmdme(model=~time+oxygen+time:oxygen,data=M,design=design)
  ##p.adjust only over interction p.values using false descovery rate method
  pInteraction <- p.values(fit,term="time:oxygen")[[1]]
  FDRIntercept <- p.adjust(fit,term="time:oxygen",method="fdr")[[1]]
  corrected <- sum(pInteraction < 0.05) - sum(FDRIntercept < 0.05)
}
```

---

permutation — permutation *of the specified Linear Model for Designed Multivariate Experiment (lmDME)*

Description

Produces de specified lmDME plus the requied permuted objects (sampling the columns), usign the same parameters to fit the additional models.

Usage

```
  ## S4 method for signature 'formula,data.frame,data.frame'
permutation(model,
    data, design, Bayes = FALSE, verbose = FALSE,
    NPermutations=100,nCpus=1, ...)
```

Arguments

model	formula object to carry out the decomposition
data	data.frame with individuals (rows) and samples/conditions (columns)
design	data.frame with the design of the experiment, (rows) samples/conditions as in data columns and as many columns to indicate the factors presents in each sample.
Bayes	Should limma estimate empirical Bayes statistics, i. e., moderated t-statistics? Default value is FALSE.
verbose	Should the process progress be printed? Default value is FALSE.
...	Additional parameters for lmFit function
NPermutations	number of permutations to be calculated. Default value is 100.
nCpus	number of cores to be used. Default value is 1, i.e. sequencial calculation.

Value

list	contains the original lmDME object plus the required amount of permuted versions

Author(s)

Cristobal Fresno and Elmer A Fernandez

See Also

lmdme

Examples

```
if (require(stemHypoxia)) {
  data(stemHypoxia)
  ##Just to make a balance dataset in the Fisher sense (2 samples per time*oxygen levels)
  design<-design[design$time %in% c("0.5","1","5") & design$oxygen %in% c("1","5","21"),]
  design$time <-as.factor(design$time)
  design$oxygen<-as.factor(design$oxygen)
  rownames(M)<-M[,1]
  M <- M[,colnames(M) %in% design$samplename] #Keeping appropiate samples only
 ##Just to test if it works. On real scenario use NPermutations >= 100 if the conditions (columns) of M allows it.
 permuted <- permutation(model=~time*oxygen,data=M,design=design, NPermutations=2,nCpus=3)##verbose=FA
}
```

---

print Show, Print *or* Summary *a lmDME object*

Description

Generic Show/Print/Summary Method for lmDME class output visualization.

Usage

```
  ## S4 method for signature 'lmDME'
print(x,term)

  ## S4 method for signature 'lmDME'
show(object)

  ## S4 method for signature 'lmDME'
summary(object)
```

Arguments

x	lmDME class object
object	lmDME class object
term	character with the corresponding term to return. Default value is NULL to return every decomposed term (if more than one available)

Value

according to the call

print, show or summary:
: console output text with increasing detail of lmDME object.

show or summary
: console output text of the lmDME object, plus a data.frame with model decomposition summary data.

Author(s)

Cristobal Fresno and Elmer A Fernandez

See Also

lmdme, coef, resid, fitted, modelDecomposition, components, componentsType

Examples

```
if (require(stemHypoxia)) {
  data(stemHypoxia)
  ##Just to make a balance dataset in the Fisher sense (2 samples per time*oxygen levels)
  design<-design[design$time %in% c("0.5","1","5") & design$oxygen %in% c("1","5","21"),]
  design$time <-as.factor(design$time)
  design$oxygen<-as.factor(design$oxygen)
  rownames(M)<-M[,1]
  M <- M[,colnames(M) %in% design$samplename] #Keeping appropiate samples only
  ##ANOVA decomposition
  fit <- lmdme(model=~time+oxygen+time:oxygen,data=M,design=design)
}
## Not run:
  fit #equivalent to call show(fit)
  print(fit)
  summary(fit)

## End(Not run)
```

---

screeplot *Plot a* screeplot *of a PCA decomposed lmDME object*

Description

Screeplot over each decomposed "pca" model present in lmDME components slot.

Usage

```
  ## S4 method for signature 'lmDME'
screeplot(x, independent=TRUE,
    col=seq(along=components(x)), npcs, term=NULL, mfcol,
    ...)
```

Arguments

x	lmDME class object.
independent	logical indicating whether the screeplots should be ploted together. Default value is FALSE.
col	which color to use for each decomposed model. Default value seq(along=components(x)).
npcs	integer with the number of components to plot. By default all present componets are ploted.
term	character with the corresponding term/s for biploting. Default value is NULL in order to obtain every available biplot/s.
mfcol	numeric vector for par layout. If missing mfcol=c(1,2) will be used if more than one biplot is available. Use mfcol==NULL to override par call inside biplot function.
...	additional parameters for screeplot or plot/lines acording to independent FALSE/TRUE respectively.

Value

ploted screeplot/s of the components slot if PCA decomposition was applied.

Author(s)

Cristobal Fresno and Elmer A Fernandez

See Also

stats::screeplot

Examples

```
if (require(stemHypoxia)) {
  data(stemHypoxia)
  ##Just to make a balance dataset in the Fisher sense (2 samples per time*oxygen levels)
  design<-design[design$time %in% c("0.5","1","5") & design$oxygen %in% c("1","5","21"),]
  design$time <-as.factor(design$time)
  design$oxygen<-as.factor(design$oxygen)
  rownames(M)<-M[,1]
  M <- M[,colnames(M) %in% design$samplename] #Keeping appropiate samples only
  ##ANOVA decomposition
  fit <- lmdme(model=~time+oxygen+time:oxygen,data=M,design=design)
 ##ASCA for all the available terms, over those subjects/genes where at least one interaction coefficient is statistically di
  id<-F.p.values(fit,term="time:oxygen")[[1]]<0.001
  decomposition(fit,decomposition="pca",scale="row", subset=id)
}
## Not run:
par(mfrow=c(2,2)); screeplot(fit,mfcol=NULL) ##Do not call par inside
screeplot(fit,term="time")##Just the term of interest
screeplot(fit,term=c("time","oxygen"),mfcol=c(1,1))##In separate graphics
screeplot(fit,mfcol=c(1,3))##All term in the same graphic
screeplot(fit,independent=FALSE)##All in the same graphic

## End(Not run)
```

Index