

Package ‘Imms’

March 7, 2016

Version 1.3.3

Date 2016-03-04

Title Linear Mixed Effect Model Splines for Modelling and Analysis of Time Course Data

Author Jasmin Straube [aut, cre],
Kim-Anh Le Cao [aut],
Emma Huang [aut],
Dominique Gorse [ctb]

Maintainer Jasmin Straube <j.straube@qfab.org>

Depends R (>= 3.0.0), ggplot2

Imports stats, methods, nlme, lmeSplines, parallel, reshape2, gdata,
gplots, gridExtra

Description Linear Mixed effect Model Splines ('Imms') implements linear mixed effect model splines for modelling and differential expression for highly dimensional data sets: investNoise() for quality control and filterNoise() for removing non-informative trajectories; ImmSpline() to model time course expression profiles and ImmsDE() performs differential expression analysis to identify differential expression between groups, time and/or group x time interaction.

License GPL (>= 2)

RoxygenNote 5.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2016-03-07 01:09:11

R topics documented:

Imms-package	2
deriv.Immspline	3
filterNoise	4
investNoise	5
kidneySimTimeGroup	7
Imms-class	8

ImmsDE	8
Immsde-class	10
lmmSpline	11
lmmSpline-class	13
noise-class	14
plot.lmmsde	14
plot.lmmspline	16
plot.noise	17
predict.lmmspline	18
summary.lmmsde	19
summary.lmmspline	19
summary.noise	20

Index 22

lmms-package	<i>Data-driven mixed effect model splines fit and differential expression analysis</i>
--------------	--

Description

The package provides quality control and filtering methods as well as linear mixed effect model splines techniques for modelling and differential expression analysis to model and mine highly dimensional data sets: `investNoise` to identify noisy profiles and `filterNoise` to remove them; `lmmSpline` to model heterogeneous time course expression profiles; `ImmsDE` to perform differential expression analysis of time course data to identify differential expression over time, between groups or time and group interaction.

Details

Package:	lmms
Type:	Package
Version:	1.3.3
Date:	2016-03-04
License:	GPL-2
LazyLoad:	yes

Functions for quality control and filtering: [investNoise](#), [filterNoise](#), [summary.noise](#), [plot.noise](#)
 Functions for data modelling: [lmmSpline](#), [lmmsDE](#), [deriv.lmmspline](#), [predict.lmmspline](#)
 Functions for summarization: [summary.lmmspline](#), [summary.lmmsde](#)
 Functions for plots: [plot.lmmspline](#), [plot.lmmsde](#)

Author(s)

Jasmin Straube with contributions from Kim-Anh Le Cao, Emma Huang and Dominique Gorse

Maintainer: Jasmin Straube <j.straube@qfab.org>

deriv.lmmspline *Derivative information for lmmspline objects*

Description

Calculates the derivative information for lmmspline objects with a "p-spline" or "cubic p-spline" basis.

Usage

```
## S3 method for class 'lmmspline'
deriv(expr, ...)
```

Arguments

expr An object of class lmmspline.
 ... Additional arguments which are passed to deriv.

Value

deriv returns an object of class lmmspline containing the following components:

predSpline	data.frame containing the predicted derivative values based on the linear model object or the linear mixed effect model object.
modelsUsed	numeric vector indicating the model used to fit the data. 0 = linear model, 1 = linear mixed effect model spline (LMMS) with defined basis ("cubic" by default), 2 = LMMS taking subject-specific random intercept, 3 = LMMS with subject specific intercept and slope.
model	list of models used to model time profiles.
derivative	logical value indicating if the predicted values are the derivative information.

Examples

```
## Not run:
data(kidneySimTimeGroup)
# run lmmSpline on the samples from group 1 only
G1 <- which(kidneySimTimeGroup$group=="G1")
testLMMSplineTG<- lmmSpline(data=kidneySimTimeGroup$data[G1,],
                             time=kidneySimTimeGroup$time[G1],
                             sampleID=kidneySimTimeGroup$sampleID[G1],
                             basis="p-spline",keepModels=T)
testLMMSplineTGDeri <- deriv(testLMMSplineTG)
summary(testLMMSplineTGDeri)
## End(Not run)
```

filterNoise	<i>Filter non-informative trajectories</i>
-------------	--

Description

Function to remove non-informative trajectories

Usage

```
filterNoise(data, noise, RTCutoff, RICutoff, propMissingCutoff, fcCutoff)
```

```
## S4 method for signature
## 'matrixOrframe,
## noise,
## missingOrnumeric,
## missingOrnumeric,
## missingOrnumeric,
## missingOrnumeric'
filterNoise(data,
  noise, RTCutoff, RICutoff, propMissingCutoff, fcCutoff)
```

Arguments

data	data.frame or matrix containing the samples as rows and features as columns.
noise	an object of class noise containing time and individual to molecule sd ratios number of missing values and maximum fold changes.
RTCutoff	numeric the R_T cutoff to remove non-informative trajectories.
RICutoff	numeric the R_I to remove non-informative trajectories.
propMissingCutoff	numeric maximum proportion of missing values in trajectories.
fcCutoff	numeric the minimum fold change observed between the mean of any two time points.

Details

filterNoise removes noisy or non-informative profiles based on selected thresholds R_I, R_T (Straube *et al.* 2015), maximum foldchanges and/or missing values.

Value

filterNoise returns an object of class list containing the following components:

data	numeric filtered data.
removedIndices	numeric removed indices

References

Straube J., Gorse A.-D., Huang B.E., Le Cao K.-A. (2015). *A linear mixed model spline framework for analyzing time course 'omics' data* PLOS ONE, 10(8), e0134540.

See Also

[investNoise](#)

Examples

```
## Not run:
data(kidneySimTimeGroup)
G1 <- kidneySimTimeGroup$group=="G1"
noiseTest <-investNoise(data=kidneySimTimeGroup$data[G1,],time=kidneySimTimeGroup$time[G1],
  sampleID=kidneySimTimeGroup$sampleID[G1])
data <-filterNoise(data=kidneySimTimeGroup$data[G1,],noise=noiseTest,RTCutoff=0.9,
  RICutoff=0.3,propMissingCutoff=0.5)$data

#Alternatively model-based clustering can be used for filtering
library(mclust)
clusterFilter <- Mclust(cbind(noiseTest@RT,noiseTest@RI),G=2)
plot(clusterFilter,what = "classification")
meanRTCluster <-tapply(noiseTest@RT,clusterFilter$classification,mean)
bestCluster <- names(meanRTCluster[which.min(meanRTCluster)])
filterdata <- kidneySimTimeGroup$data[G1,clusterFilter$classification==bestCluster]

## End(Not run)
```

investNoise

Quality control for time course profiles

Description

Function to calculate filter ratios of trajectories.

Usage

```
investNoise(data, time, sampleID, log, numCores)
```

Arguments

data	data.frame or matrix containing the samples as rows and features as columns
time	numeric vector containing the sample time point information.
sampleID	character, numeric or factor vector containing information about the unique identity of each sample
log	logical indicating log transformation of the data. Default value is TRUE

numCores alternative numeric value indicating the number of CPU cores to be used for parallelization. Default value is automatically estimated.

Details

investNoise calculates filter ratios R_T and R_I based on the time, individual and overall standard deviation as proposed by Straube *et al.* 2015.

Value

investNoise returns an object of class noise containing the following components:

name	character	the colnames or the index.
RT	numeric	the time to molecule sd ratio of each trajectory.
RI	numeric	the individual to molecule sd ratio of each trajectory.
propMissing	numeric	Proportion of missing values for each trajectory.
foldChange	numeric	the maximum absolute fold change (either for log transformed data $\max(\text{time}) - \min(\text{time})$ or not log transformed data $\max(\text{time})/\min(\text{time})$) observed between the mean of any two time points.

References

Straube J., Gorse D., Huang B.E., Le Cao K.-A. (2015). *A linear mixed model spline framework for analyzing time course 'omics' data* PLOS ONE, 10(8), e0134540.

See Also

[summary.noise](#), [plot.noise](#), [filterNoise](#)

Examples

```
## Not run:
data(kidneySimTimeGroup)
G1 <- kidneySimTimeGroup$group=="G1"
noiseTest <-investNoise(data=kidneySimTimeGroup$data[G1,],time=kidneySimTimeGroup$time[G1],
                        sampleID=kidneySimTimeGroup$sampleID[G1])
summary(noiseTest)
plot(noiseTest,colorBy="propMissing")
## End(Not run)
```

kidneySimTimeGroup *Kidney Simulation Data*

Description

This data set contains the simulated expression of 140 proteins in 40 samples from either group 1 or group 2 measured on the time points 0, 0.5, 1, 2, 3, 4.

Usage

```
data(kidneySimTimeGroup)
```

Format

A list containing the following components:

`data` data matrix with 192 rows and 140 columns. Each row represents an experimental sample, and each column a single protein.

`time` a numeric vector containing the time points on which each sample is measured

`sampleID` a character vector containing the biological replicate information of each sample

`group` a character vector indicating the group of each sample

Details

This simulated data set consists of 40 samples and 140 proteins and was based on an the existing study from Freue *et al.* (2010). Samples were measured on maximum 6 time points: 0, 0.5, 1, 2, 3, 4. Some samples have missing time points. 50 molecules were randomly selected to introduce a fold change of $\log(2)$.

Source

The Kidney Simulation Data is based on the the paper of Freue *et al.* (2010).

References

Freue, G. V. C. et al. (2010). Proteomic signatures in plasma during early acute renal allograft rejection. *Molecular & cellular proteomics*, **9**, 1954-67.

lms-class	<i>lms class a S4 superclass to extend lmspline and lmsde class.</i>
-----------	--

Description

lms class is a superclass for classes lmspline and lmsde. These classes inherit common slots.

Slots

basis An object of class character describing the basis used for modelling.

knots An object of class numeric, describing the boundaries of the splines. If not defined or if basis='cubic' knots are automatically estimated using Ruppert 2002 or are the design points when using 'cubic'.

errorMolecules Vector of class character, describing the molecules that could not be modelled.

lmsDE	<i>Differential expression analysis using linear mixed effect model splines.</i>
-------	--

Description

Function to fit a linear mixed effect model splines to perform differential expression analysis. The [lmsDE](#) function fits LMM models with either a cubic, p-spline or cubic p-spline basis and compares the models to the null models. The type of basis to use is specified with the basis argument.

Usage

```
lmsDE(data, time, sampleID, group, type,
       experiment, basis, knots, keepModels, numCores)
```

Arguments

data	data.frame or matrix containing the samples as rows and features as columns
time	numeric vector containing the sample time point information.
sampleID	character, numeric or factor vector containing information about the unique identity of each sample
group	character, numeric or factor vector containing information about the group (or class) of each sample
type	character indicating what type of analysis is to be performed. Options are "time" to identify differential expression over time, "group" to identify profiles with different baseline levels (intercepts), and "time*group" an interaction between these two . Use "all" to calculate all three types.

experiment	character describing the experiment performed for correlation handling. Use "all" for data-driven selection of model; "timecourse" for replicated experiments with less variation in individual expression values (e.g. model organism, cell culture), "longitudinal1" for different intercepts and "longitudinal2" for different intercepts and slopes.
basis	character string. What type of basis to use, matching one of "cubic" smoothing spline as defined by Verbyla <i>et al.</i> 1999, "p-spline" Durban <i>et al.</i> 2005 or a "cubic p-spline".
knots	can take an integer value corresponding to the number of knots for the chosen basis or by default calculated as in Ruppert 2002. Not in use for the 'cubic' smoothing spline basis.
keepModels	alternative logical value if you want to keep the model output. Default value is FALSE
numCores	alternative numeric value indicating the number of CPU cores to be used for parallelization. Default value is automatically estimated.

Details

ImmsDE extends the LMMS modelling framework to permit tests between groups, across time, and for interactions between the two implemented as described in Straube *et al.* 2015.

Value

ImmsDE returns an object of class `Immsde` containing the following components:

DE	data.frame returning p-values and adjusted p-values using Benjamini-Hochberg correction for multiple testing of the differential expression testing over time, group or their interaction.
modelsUsed	numeric vector indicating the model used to fit the data. 1=linear mixed effect model spline (LMMS) with defined basis ('cubic' by default) 2 = LMMS taking subject-specific random intercept, 3 = LMMS with subject specific intercept and slope.
predTime	data.frame containing predicted values based on linear model object or linear mixed effect model object.
predGroup	data.frame containing predicted values based on linear model object or linear mixed effect model object.
predTime	data.frame containing predicted values based on linear model object or linear mixed effect model object.
predTimeGroup	data.frame containing predicted for the time*group model values based on linear model object or linear mixed effect model object.
modelTime	a list of class <code>lme</code> , containing the models for every feature modelling the time effect.
modelGroup	a list of class <code>lme</code> , containing the models for every feature modelling group effect.
modelTimeGroup	a list of class <code>lme</code> , containing the models for every feature modelling time and group interaction effect.

type	an object of class character, describing the test performed either time, group, time*group or all.
experiment	an object of class character describing the model used to perform differential expression analysis.

References

Durban, M., Harezlak, J., Wand, M. P., & Carroll, R. J. (2005). *Simple fitting of subject-specific curves for longitudinal data*. *Stat. Med.*, 24(8), 1153-67.

Ruppert, D. (2002). *Selecting the number of knots for penalized splines*. *J. Comp. Graph. Stat.* 11, 735-757

Verbyla, A. P., Cullis, B. R., & Kenward, M. G. (1999). *The analysis of designed experiments and longitudinal data by using smoothing splines*. *Appl. Statist.* 18(3), 269-311.

Straube J., Gorse A.-D., Huang B.E., & Le Cao K.-A. (2015). *A linear mixed model spline framework for analyzing time course 'omics' data* PLOS ONE, 10(8), e0134540.

See Also

[summary.lmmsde](#), [plot.lmmsde](#)

Examples

```
## Not run:
data(kidneySimTimeGroup)
lmmsDEtest <-lmmsDE(data=kidneySimTimeGroup$data,time=kidneySimTimeGroup$time,
                    sampleID=kidneySimTimeGroup$sampleID,group=kidneySimTimeGroup$group)
summary(lmmsDEtest)
## End(Not run)
```

lmmsde-class

lmmsde class a S4 class that extends lmms class.

Description

lmmsde class inherits from class lmms and extends it with six further slots: DE, model.time, model.group, model.time.group, type and experiment. The class lmmsde is returned when applying [lmmsDE](#) method.

Slots

DE A data.frame returning p-values and adjusted p-values using Benjamini-Hochberg correction for multiple testing of the differential expression testing over time, group or their interaction.

modelsUsed A list of lme, containing the models used to model the particular condition of interest.

predTime A matrix returning the predicted time fit.

predGroup A matrix returning the predicted group fit.

predTimeGroup A matrix returning the predicted time group interaction fit.
 modelTime A list of class `lme`, containing the models for every molecule modelling the time effect.
 modelGroup A list of class `lme`, containing the models for every molecule modelling group effect.
 modelTimeGroup A list of class `lme`, containing the models for every molecule modelling time and group interaction effect.
 type An object of class character, describing the test performed.
 experiment An object of class character describing the model used to perform differential expression analysis.

 ImmSpline

Data-driven linear mixed effect model spline modelling

Description

Function that models a linear or linear mixed model depending on the best fit. Alternatively, the function can return THE derivation information of the fitted models for the fixed (original) times points and a chosen basis.

Usage

```
ImmSpline(data, time, sampleID, timePredict, deri, basis, knots, keepModels,numCores)
```

Arguments

data	data.frame or matrix containing the samples as rows and features as columns
time	numeric vector containing the sample time point information.
sampleID	character, numeric or factor vector containing information about the unique identity of each sample
timePredict	numeric vector containing the time points to be predicted. By default set to the original time points observed in the experiment.
deri	logical value. If TRUE returns the predicted derivative information on the observed time points. By default set to FALSE.
basis	character string. What type of basis to use, matching one of "cubic", "p-spline" or "cubic p-spline". The "cubic" basis (default) is the cubic smoothing spline as defined by Verbyla <i>et al.</i> 1999, the "p-spline" is the truncated p-spline basis as defined by Durban <i>et al.</i> 2005.
knots	Alternatively an integer, the number of knots used for the "p-spline" or "cubic p-spline" basis calculation. Otherwise calculated as proposed by Ruppert 2002. Not used for the "cubic" smoothing spline basis as it used the inner design points.
keepModels	alternative logical value if you want to keep the model output. Default value is FALSE
numCores	Alternative numeric value indicating the number of CPU cores to be used. Default value is automatically estimated.

Details

The first model (modelsUsed=0) assumes the response is a straight line not affected by individual variation.

Let $y_{ij}(t_{ij})$ be the expression of a feature for individual (or biological replicate) i at time t_{ij} , where $i = 1, 2, \dots, n$, $j = 1, 2, \dots, m_i$, n is the sample size and m_i is the number of observations for individual i for the given feature. We fit a simple linear regression of expression $y_{ij}(t_{ij})$ on time t_{ij} . The intercept β_0 and slope β_1 are estimated via ordinary least squares: $y_{ij}(t_{ij}) = \beta_0 + \beta_1 t_{ij} + \epsilon_{ij}$, where $\epsilon_{ij} \sim N(0, \sigma_\epsilon^2)$. The second model (modelsUsed=1) is nonlinear where the straight line in regression replaced with a curve modelled using here for example a spline truncated line basis (basis="p-spline") as proposed Durban *et al.* 2005:

$$y_{ij}(t_{ij}) = f(t_{ij}) + \epsilon_{ij},$$

where $\epsilon_{ij} \sim N(0, \sigma_\epsilon^2)$.

The penalized spline is represented by f , which depends on a set of knot positions $\kappa_1, \dots, \kappa_K$ in the range of t_{ij} , some unknown coefficients u_k , an intercept β_0 and a slope β_1 . The first term in the above equation can therefore be expanded as:

$$f(t_{ij}) = \beta_0 + \beta_1 t_{ij} + \sum_{k=1}^K u_k (t_{ij} - \kappa_k)_+,$$

with $(t_{ij} - \kappa_k)_+ = t_{ij} - \kappa_k$, if $t_{ij} - \kappa_k > 0$, 0 otherwise.

The choice of the number of knots K and their positions influences the flexibility of the curve. If the argument knots=missing, we use a method proposed by Ruppert 2002 to estimate the number of knots given the measured number of time points T , so that the knots $\kappa_1 \dots \kappa_K$ are placed at quantiles of the time interval of interest:

$$K = \max(5, \min(\text{floor}(\frac{T}{4}), 40)).$$

In order to account for individual variation, our third model (modelsUsed=2) adds a subject-specific random effect U_i to the mean response $f(t_{ij})$. Assuming $f(t_{ij})$ to be a fixed (yet unknown) population curve, U_i is treated as a random realization of an underlying Gaussian process with zero-mean and variance σ_U^2 and is independent from the random error ϵ_{ij} :

$$y_{ij}(t_{ij}) = f(t_{ij}) + U_i + \epsilon_{ij}$$

with $U_i \sim N(0, \sigma_U^2)$ and $\epsilon_{ij} \sim N(0, \sigma_\epsilon^2)$. In the equation above, the individual curves are expected to be parallel to the mean curve as we assume the individual expression curves to be constant over time. A simple extension to this model is to assume individual deviations are straight lines. The fourth model (modelsUsed=3) therefore fits individual-specific random intercepts a_{i0} and slopes a_{i1} :

$$y_{ij}(t_{ij}) = f(t_{ij}) + a_{i0} + a_{i1} t_{ij} + \epsilon_{ij}$$

with $\epsilon_{ij} \sim N(0, \sigma_\epsilon^2)$ and $(a_{i0}, a_{i1})^T \sim N(0, \Sigma)$. We assume independence between the random intercept and slope. @return ImmSpline returns an object of class ImmSpline containing the following components:

- predSplinedata.frame containing predicted values based on linear model object or linear mixed effect model object.
- modelsUsednumeric vector indicating the model used to fit the data. 0 = linear model, 1=linear mixed effect model spline (LMMS) with defined basis ('cubic' by default) 2 = LMMS taking subject-specific random intercept, 3 = LMMS with subject specific intercept and slope.
- modellist of models used to model time profiles.
- derivativelogical value indicating if the predicted values are the derivative information.

References

- Durban, M., Harezlak, J., Wand, M. P., & Carroll, R. J. (2005). *Simple fitting of subject-specific curves for longitudinal data*. Stat. Med., 24(8), 1153-67.
- Ruppert, D. (2002). *Selecting the number of knots for penalized splines*. J. Comp. Graph. Stat. 11, 735-757
- Verbyla, A. P., Cullis, B. R., & Kenward, M. G. (1999). *The analysis of designed experiments and longitudinal data by using smoothing splines*. Appl.Statist, 18(3), 269-311.
- Straube J., Gorse A.-D., Huang B.E., Le Cao K.-A. (2015). *A linear mixed model spline framework for analyzing time course 'omics' data* PLOS ONE, 10(8), e0134540.

See Also

[summary.lmmspline](#), [plot.lmmspline](#), [predict.lmmspline](#), [deriv.lmmspline](#)

Examples

```
## Not run:
data(kidneySimTimeGroup)
# running for samples in group 1
G1 <- which(kidneySimTimeGroup$group=="G1")
testLMMSpline<- lmmSpline(data=kidneySimTimeGroup$data[G1,],time=kidneySimTimeGroup$time[G1],
                          sampleID=kidneySimTimeGroup$sampleID[G1])
summary(testLMMSpline)
DerivTestLMMSplineTG<- lmmSpline(data=as.data.frame(kidneySimTimeGroup$data[G1,]),
                                  time=kidneySimTimeGroup$time[G1],sampleID=kidneySimTimeGroup$sampleID[G1],
                                  deri=TRUE,basis="p-spline")
summary(DerivTestLMMSplineTG)
## End(Not run)
```

lmmspline-class

lmmspline class a S4 class that extends lmmms class.

Description

lmmspline class inherits from class lmmms and extends it with three further slots: predSpline, modelsUsed, models. The class is returned when applying [lmmmspline](#) method.

Slots

predSpline A data.frame returning the fitted values for the time points of interest.

models A list of class `lm` or `lme` containing the models for every molecule

modelsUsed A list of class `lm` or `lme`, containing the models used to model the particular feature of interest.

derivative A logical value indicating if the derivative was calculated.

noise-class	noise <i>S4</i> class
-------------	-----------------------

Description

The class `noise` is returned when applying `investNoise` method.

Slots

name character vector. The name of the trajectory.

RT A numeric vector, containing the time to molecule standard deviation ratios for every trajectory.

RI A numeric vector, containing the individual to molecule standard deviation ratios for every trajectory.

propMissing A numeric vector, containing the proportion of missing values for every trajectory.

foldChange A numeric vector, containing the maximum fold change of the mean between any two time points.

plot.lmmsde	<i>Plot of lmmsde objects</i>
-------------	-------------------------------

Description

Plot of the raw data the mean and the fitted `lmmsde` profile.

Usage

```
## S3 method for class 'lmmsde'
plot(x, y, data, time, group, type, smooth, mean, ...)
```

Arguments

x	An object of class <code>lmmsde</code> .
y	numeric or character value. Either the row index or the row name determining which feature should be plotted.
data	alternative matrix or data.frame containing the original data for visualisation purposes.
time	alternative numeric indicating the sample time point. Vector of same length as row length of data for visualisation purposes.
group	alternative numeric indicating the sample group. Vector of same length as row length of data for visualisation purposes.
type	a character indicating what model to plot. Default 'all', options: 'time', 'group', 'group*time'.
smooth	an optional logical value. By default set to FALSE. If TRUE smooth representation of the fitted values.
mean	alternative logical if the mean should be displayed. By default set to TRUE.
...	Additional arguments which are passed to plot.

Value

plot showing raw data, mean profile and fitted profile.

Examples

```
## Not run:
data(kidneySimTimeGroup)
lmmsDEtest11 <-lmmsDE(data=kidneySimTimeGroup$data,time=kidneySimTimeGroup$time,
                    sampleID=kidneySimTimeGroup$sampleID,
                    group=kidneySimTimeGroup$group,
                    experiment="longitudinal1",basis="p-spline",keepModels=T)
plot(lmmsDEtest11,y=2,type="all")
plot(lmmsDEtest11,y=2,type="time")
plot(lmmsDEtest11,y=2,type="group")
plot(lmmsDEtest11,y=2,type="group*time",smooth=TRUE)

#to save memory do not keep the models
lmmsDEtest11 <-lmmsDE(data=kidneySimTimeGroup$data,time=kidneySimTimeGroup$time,
                    sampleID=kidneySimTimeGroup$sampleID,
                    group=kidneySimTimeGroup$group,
                    experiment="longitudinal1",basis="p-spline",keepModels=F)
# just the fitted trajectory
plot(lmmsDEtest11,y=2,type="all")

plot(lmmsDEtest11,y=2,type="all",data=kidneySimTimeGroup$data,time=kidneySimTimeGroup$time,
     group=kidneySimTimeGroup$group)
## End(Not run)
```

plot.lmmspline *Plot of lmmspline object*

Description

Plots the raw data, the mean and the fitted or derivative information of the lmmspline object.

Usage

```
## S3 method for class 'lmmspline'
plot(x, y, data, time, smooth, mean, ...)
```

Arguments

x	An object of class lmmspline.
y	character or numeric value. Determining which feature should be plotted can be either the index or the name of the feature.
data	alternative matrix or data.frame containing the original data for visualisation purposes.
time	alternative numeric indicating the sample time point. Vector of same length as row length of data for visualisation purposes.
smooth	an optional logical value. Default FALSE, if TRUE smooth representation of the fitted values.
mean	alternative logical if the mean should be displayed. By default set to TRUE.
...	Additional arguments which are passed to plot.

Value

plot showing raw data, mean profile and fitted profile.

Examples

```
## Not run:
data(kidneySimTimeGroup)
# running for samples in group 1
G1 <- which(kidneySimTimeGroup$group=="G1")
testLmmspline <- lmmSpline(data=kidneySimTimeGroup$data[G1,],
                          time=kidneySimTimeGroup$time[G1],
                          sampleID=kidneySimTimeGroup$sampleID[G1],keepModels=T)

plot(testLmmspline, y=2)
plot(testLmmspline, y=2, smooth=TRUE)
# Don't keep the models to improve memory usage
testLmmspline <- lmmSpline(data=kidneySimTimeGroup$data[G1,],
                          time=kidneySimTimeGroup$time[G1],
                          sampleID=kidneySimTimeGroup$sampleID[G1],keepModels=F)
```



```

#plot only the fitted values
plot(testLmmspline, y=2)
#plot fitted values with original data
plot(testLmmspline, y=2, data=kidneySimTimeGroup$data[G1,], time=kidneySimTimeGroup$time[G1])

## End(Not run)

```

plot.noise	<i>Plot of associations objects</i>
------------	-------------------------------------

Description

Plot of the filter ratios R_T and R_I as proposed by Straube et al 2014.

Usage

```

## S3 method for class 'noise'
plot(x, colorBy = "propMissing", fcCutoff = NA,
     propMissingCutoff = NA, ...)

```

Arguments

x	an object of class matrix or data.frame.
colorBy	associations the variable to be colored by. Default 'propMissing', options: 'propMissing','fc'.
fcCutoff	an optional numeric value to remove ratios with low fold changes.
propMissingCutoff	an optional numeric value to remove ratios with high number of missing values.
...	ignored

Value

plot showing filter ratios R_T and R_I as proposed by Straube *et al.* 2014. Filter ratios can either be colored by proportion of missing values or maximum fold change.

References

Straube J., Gorse D., Huang B.E., Le Cao K.-A.(2014). *A linear mixed model spline framework for analyzing time course 'omics' data* Submitted

See Also

[investNoise](#), [filterNoise](#)

Examples

```
## Not run:
data(kidneySimTimeGroup)
G1 <- kidneySimTimeGroup$group=="G1"
noiseTest <- investNoise(data=kidneySimTimeGroup$data[G1,], time=kidneySimTimeGroup$time[G1],
  sampleID=kidneySimTimeGroup$sampleID[G1])
plot(noiseTest, colorBy="fc")

## End(Not run)
```

predict.lmmspline *Predicts fitted values of an lmmspline Object*

Description

Predicts the fitted values of an lmmspline object for time points of interest.

Usage

```
## S3 method for class 'lmmspline'
predict(object, timePredict, numCores, ...)
```

Arguments

object	an object inheriting from class lmmspline.
timePredict	an optional numeric vector. Vector of time points to predict fitted values. If missing uses design points.
numCores	alternative numeric value indicating the number of CPU cores to be used for parallelization. By default estimated automatically.
...	ignored.

Value

matrix containing predicted values for the requested time points from argument timePredict.

Examples

```
## Not run:
data(kidneySimTimeGroup)
G1 <- which(kidneySimTimeGroup$group=="G1")
testLMMSpline<- lmmSpline(data=kidneySimTimeGroup$data[G1,],
  time=kidneySimTimeGroup$time[G1],
  sampleID=kidneySimTimeGroup$sampleID[G1], keepModels=T)
mat.predict <- predict(testLMMSpline, timePredict=c(seq(1,4, by=0.5)))
## End(Not run)
```

summary.lmmsde	<i>Summary of a lmmsde Object</i>
----------------	-----------------------------------

Description

Summarizes the lmmsde object returned by the [lmmsDE](#) method. Including the models fitted, parameter used and the number of features declared as differentially expressed.

Usage

```
## S3 method for class 'lmmsde'
summary(object, ...)
```

Arguments

object	An object of class lmmsde .
...	Additional arguments which are passed to summary.

Value

summary of the lmmsde object.

Examples

```
## Not run:
data(kidneySimTimeGroup)
lmmsDEtest <-lmmsDE(data=kidneySimTimeGroup$data,time=kidneySimTimeGroup$time,
  sampleID=kidneySimTimeGroup$sampleID,group=kidneySimTimeGroup$group)
summary(lmmsDEtest)
## End(Not run)
```

summary.lmmspline	<i>Summary of a lmmspline Object</i>
-------------------	--------------------------------------

Description

Summarizes the lmmspline object returned by the [lmmSpline](#) method. Including the models fitted and parameter used.

Usage

```
## S3 method for class 'lmmspline'
summary(object, ...)
```

Arguments

object An object of class lmspline.
 ... Additional arguments which are passed to summary.

Value

Summary of the lmspline object.

Examples

```
## Not run:
data(kidneySimTimeGroup)
# running for samples in group 1
G1 <- which(kidneySimTimeGroup$group=="G1")
testLMSplineTG<- lmspline(data=kidneySimTimeGroup$data[G1,],
                          time=kidneySimTimeGroup$time[G1],
                          sampleID=kidneySimTimeGroup$sampleID[G1])
summary(testLMSplineTG)
## End(Not run)
```

summary.noise

Summary of a noise Object

Description

Summarizes the noise object returned by the [investNoise](#) method.

Usage

```
## S3 method for class 'noise'
summary(object, ...)
```

Arguments

object An object of class noise.
 ... ignored

Value

Summary of the noise object.

Examples

```
## Not run:
data(kidneySimTimeGroup)
# running for samples in group 1
G1 <- which(kidneySimTimeGroup$group=="G1")
noiseTest<- investNoise(data=kidneySimTimeGroup$data[G1,],
                        time=kidneySimTimeGroup$time[G1],
                        sampleID=kidneySimTimeGroup$sampleID[G1])
summary(noiseTest)
## End(Not run)
```

Index

*Topic **datasets**

kidneySimTimeGroup, 7

*Topic **package**

lmms-package, 2

deriv.lmmspline, 2, 3, 13

filterNoise, 2, 4, 6, 17

filterNoise, matrixOrframe, noise, missingOrnumeric, missingOrnumeric, missingOrnumeric, missingOrnumeric-met
(filterNoise), 4

investNoise, 2, 5, 5, 14, 17, 20

kidneySimTimeGroup, 7

lm, 14

lme, 9, 11, 14

lmms (lmms-package), 2

lmms-class, 8

lmms-package, 2

lmmsDE, 2, 8, 8, 10, 19

lmmsde-class, 10

lmmspline, 2, 11, 13, 19

lmmspline-class, 13

noise-class, 14

plot.lmmsde, 2, 10, 14

plot.lmmspline, 2, 13, 16

plot.noise, 2, 6, 17

predict.lmmspline, 2, 13, 18

summary.lmmsde, 2, 10, 19

summary.lmmspline, 2, 13, 19

summary.noise, 2, 6, 20