Package ‘latrend’

November 18, 2020

Type Package
Version 1.0.1
Date 2020-11-12
Title Framework for Clustering Longitudinal Data
Description A framework for clustering longitudinal datasets in a standardized way. Provides an interface to existing R packages for clustering longitudinal univariate trajectories, facilitating reproducible and transparent analyses. Additionally, standard tools are provided to support cluster analyses, including repeated estimation, model validation, and model assessment. The interface enables users to compare results between methods, and to implement and evaluate new methods with ease.

Maintainer Niek Den Teuling <niek.den.teuling@philips.com>

URL https://github.com/philips-software/latrend

BugReports https://github.com/philips-software/latrend/issues

License GPL (>= 2)

Encoding UTF-8

LazyData true

Depends R (>= 3.6.0)

Imports stats (>= 3.6.0), methods (>= 3.6.0), R.utils (>= 2.9.0), assertthat (>= 0.2.1), foreach (>= 1.4.7), data.table (>= 1.12.0), longitudinalData (>= 2.4.1), magrittr (>= 1.5), plyr (>= 1.8.4), ggplot2 (>= 2.2.1), matrixStats (>= 0.55.0), stackoverflow (>= 0.3.0), clusterCrit (>= 1.2.8), mclustcomp (>= 0.3.1), scales (>= 1.0.0), caret (>= 6.0), lme4 (>= 1.1), mclust (>= 5.4.5)

Suggests testthat (>= 3.0.0), roxygen2 (>= 7.1.0), knitr (>= 1.24), rmarkdown (>= 1.18), kml (>= 2.4.1), lcm (>= 1.7.8), longclust (>= 1.2.3), mixtools (>= 1.1.0), flexmix (>= 2.3), fda (>= 2.4.8), funFEM (>= 1.1), igraph (>= 1.2.4), crimCV (>= 0.9.6), akmedoids (>= 0.1.5), dtwclust (>= 5.5.6), mixAK (>= 5.3), psych (>= 1.9.12.31), qqplotr (>= 0.0.4)

RoxygenNote 7.1.1
VignetteBuilder knitr

Collate 'assert.R' 'compute.R' 'data.R' 'formula.R' 'generics.R'
'latrend.R' 'make.R' 'matrix.R' 'method.R' 'methodMatrix.R'
'methodAKMedoids.R' 'methodCrimCV.R' 'methodCustom.R'
'methodDttwclust.R' 'methodFlexmix.R' 'methodFlexmixGBTM.R'
'methodFunFEM.R' 'plot.R' 'model.R' 'modelCustom.R'
'methodTwoStep.R' 'methodGCKM.R' 'methodKML.R' 'methodLMKM.R'
'methodLcmmGMM.R' 'methodLcmmGBTM.R' 'methodLongclust.R'
'methodMclustLLPA.R' 'methodMixAK_GLMM.R' 'methodMixTVEM.R'
'methodMixtoolsGMM.R' 'methodMixtoolsNPRM.R' 'methodRandom.R'
'methodStratify.R' 'methods.R' 'metrics.R' 'model-summary.R'
'model-transform.R' 'modelApprox.R' 'modelCrimCV.R'
'modelDttwclust.R' 'modelFlexmix.R' 'modelFunFEM.R' 'modelKML.R'
'modelLMKM.R' 'modelLcmmGMM.R' 'modelLcmmGBTM.R'
'modelLongclust.R' 'modelMclustLLPA.R' 'modelMixAK_GLMM.R'
'modelMixAK_GLMMList.R' 'modelMixTVEM.R' 'modelMixtoolsGMM.R'
'modelMixtoolsR.R' 'modelPartition.R' 'modelStratify.R'
'modelTwoStep.R' 'modelWeightedPartition.R' 'models.R'
'random.R' 'verbose.R' 'zzz.R'

NeedsCompilation no

Author Niek Den Teuling [aut, cre] (<https://orcid.org/0000-0003-1026-5080>),
Steffen Pauws [ctb],
Edwin van den Heuvel [ctb],
Copyright © 2020 Koninklijke Philips N.V. [cph]

Repository CRAN

Date/Publication 2020-11-18 10:10:02 UTC

R topics documented:

latrend-package .................................................. 5
as.data.frame.lcMethod .......................................... 6
as.data.frame.lcMethods ......................................... 6
as.data.frame.lcModels ......................................... 7
as.lcMethods ...................................................... 8
as.lcModels ....................................................... 8
as.list.lcMethod .................................................. 9
clustarNames .................................................... 10
clustarNames<- .................................................. 10
clustarProportions .............................................. 11
clustarSizes ...................................................... 11
clustarTrajectories ............................................. 12
coeIc.lcModel .................................................... 13
coeIcMatrix ...................................................... 13
covarizedMatrix .................................................. 13
createTestDataFold .............................................. 14
createTestDataFolds ............................................ 15
topics documented:

- createTrainDataFolds .......................... 16
- dcastRepeatedMeasures ......................... 17
- defineExternalMetric .......................... 17
- defineInternalMetric ........................... 18
- deviance.lcModel .............................. 19
- df.residual.lcModel ............................ 19
- estimationTime ................................. 20
- evaluate.lcMethod .............................. 20
- externalMetric.lcModel.lcModel-method ..... 21
- fitted.lcModel .................................. 22
- formula.lcMethod ............................... 23
- formula.lcModel ................................ 24
- generateLongData ................................ 24
- getExternalMetricDefinition .................. 25
- getExternalMetricNames ....................... 26
- getInternalMetricDefinition .................. 26
- getInternalMetricNames ....................... 27
- getLcMethod .................................... 27
- ids .............................................. 28
- idVariable ..................................... 28
- latrend ........................................ 29
- latrendBatch ................................... 30
- latrendBoot ................................... 31
- latrendCV ...................................... 32
- latrendData .................................... 33
- latrendRep ..................................... 34
- lcApproxModel-class ............................ 35
- lcMethod ....................................... 36
- lcMethod-class .................................. 36
- lcMethod.call ................................... 39
- lcMethodAKMedoids .............................. 40
- lcMethodCrimCV ................................ 41
- lcMethodCustom ................................ 42
- lcMethodDtwclust ................................ 43
- lcMethodFlexmix ................................ 44
- lcMethodFlexmixGBTM ............................ 45
- lcMethodFunFEM ................................ 46
- lcMethodGCKM ................................... 47
- lcMethodKML .................................... 48
- lcMethodLcmmGBTM .............................. 49
- lcMethodLcmmGMM ............................... 50
- lcMethodLMKM ................................... 51
- lcMethodLongclust ................................ 52
- lcMethodMclustLLPA ............................. 53
- lcMethodMixAK_GLMM ............................ 54
- lcMethodMixtoolsGMM ............................ 55
- lcMethodMixtoolsNPRM ........................... 56
- lcMethodMixTVEM ................................ 57
<table>
<thead>
<tr>
<th>R topics documented</th>
</tr>
</thead>
<tbody>
<tr>
<td>lcMethodRandom</td>
</tr>
<tr>
<td>lcMethods</td>
</tr>
<tr>
<td>lcMethodStratify</td>
</tr>
<tr>
<td>lcMethodTwoStep</td>
</tr>
<tr>
<td>lcModel-class</td>
</tr>
<tr>
<td>lcModelCustom</td>
</tr>
<tr>
<td>lcModelPartition</td>
</tr>
<tr>
<td>lcModels</td>
</tr>
<tr>
<td>lcModelWeightedPartition</td>
</tr>
<tr>
<td>logLik.lcModel</td>
</tr>
<tr>
<td>max.lcModels</td>
</tr>
<tr>
<td>meltRepeatedMeasures</td>
</tr>
<tr>
<td>metric</td>
</tr>
<tr>
<td>min.lcModels</td>
</tr>
<tr>
<td>model.data.lcModel</td>
</tr>
<tr>
<td>model.frame.lcModel</td>
</tr>
<tr>
<td>nClusters</td>
</tr>
<tr>
<td>nIds</td>
</tr>
<tr>
<td>nobs.lcModel</td>
</tr>
<tr>
<td>plot.lcModel,ANY-method</td>
</tr>
<tr>
<td>plotClusterTrajectories</td>
</tr>
<tr>
<td>plotMetric</td>
</tr>
<tr>
<td>plotTrajectories</td>
</tr>
<tr>
<td>postprob</td>
</tr>
<tr>
<td>postprobFromAssignments</td>
</tr>
<tr>
<td>predict.lcModel</td>
</tr>
<tr>
<td>predictAssignments</td>
</tr>
<tr>
<td>predictForCluster</td>
</tr>
<tr>
<td>predictPostprob</td>
</tr>
<tr>
<td>print.lcMethod</td>
</tr>
<tr>
<td>print.lcModels</td>
</tr>
<tr>
<td>qqPlot</td>
</tr>
<tr>
<td>residuals.lcModel</td>
</tr>
<tr>
<td>responseVariable</td>
</tr>
<tr>
<td>sigma.lcModel</td>
</tr>
<tr>
<td>strip</td>
</tr>
<tr>
<td>subset.lcModels</td>
</tr>
<tr>
<td>summary.lcModel</td>
</tr>
<tr>
<td>time.lcModel</td>
</tr>
<tr>
<td>timeVariable</td>
</tr>
<tr>
<td>trajectories</td>
</tr>
<tr>
<td>trajectoryAssignments</td>
</tr>
<tr>
<td>transformFitted</td>
</tr>
<tr>
<td>transformLatrendData</td>
</tr>
<tr>
<td>transformPredict</td>
</tr>
<tr>
<td>update.lcMethod</td>
</tr>
<tr>
<td>update.lcModel</td>
</tr>
<tr>
<td>which.weight</td>
</tr>
</tbody>
</table>
Description

A framework for clustering longitudinal datasets in a standardized way. Provides an interface to existing R packages for clustering longitudinal univariate trajectories, facilitating reproducible and transparent analyses. Additionally, standard tools are provided to support cluster analyses, including repeated estimation, model validation, and model assessment. The interface enables users to compare results between methods, and to implement and evaluate new methods with ease.

Getting started

- See vignette("demo",package = "latrend") for an introduction to conducting a longitudinal cluster analysis on a example case study.
- See vignette("custom",package = "latrend") for examples on constructing your own cluster models.
- See vignette("validation",package = "latrend") for examples on applying internal cluster validation.

Author(s)

Maintainer: Niek Den Teuling <niek.den.teuling@philips.com> (ORCID)

Other contributors:

- Steffen Pauws <s.c.pauws@uvt.nl> [contributor]
- Edwin van den Heuvel <e.r.v.d.heuvel@tue.nl> [contributor]
- Copyright © 2020 Koninklijke Philips N.V. [copyright holder]

See Also

Useful links:

- https://github.com/philips-software/latrend
- Report bugs at https://github.com/philips-software/latrend/issues
as.data.frame.lcMethod

Convert lcMethod arguments to a list of atomic types

Description

Converts the arguments of a lcMethod to a named list of atomic types.

Usage

## S3 method for class 'lcMethod'
as.data.frame(x, ..., eval = FALSE, nullValue = NA, envir = NULL)

Arguments

- **x**: lcMethod to be coerced to a character vector.
- **...**: Additional arguments.
- **eval**: Whether to evaluate the arguments in order to replace expression if the resulting value is of a class specified in evalClasses.
- **nullValue**: Value to use to represent the NULL type. Must be of length 1.
- **envir**: The environment in which to evaluate the arguments. If NULL, the environment associated with the object is used. If not available, the parent.frame() is used.

Value

A single-row data.frame where each columns represents an argument call or evaluation.

See Also

Other lcMethod functions: \[[, lcMethod-method, as.data.frame.lcMethods(), as.lcMethods(),
as.list.lcMethod(), evaluate.lcMethod(), formula.lcMethod(), lcMethod-class, update.lcMethod()\]

as.data.frame.lcMethods

Convert a list of lcMethod objects to a data.frame

Description

Converts a list of lcMethod objects to a data.frame.

Usage

## S3 method for class 'lcMethods'
as.data.frame(x, ..., eval = FALSE, nullValue = NA, envir = NULL)
as.data.frame.lcModels

Arguments

x
   the lcMethods or list to be coerced to a data.frame.

... Additional arguments.
eval Whether to evaluate the arguments in order to replace expression if the resulting value is of a class specified in evalClasses.
nullValue Value to use to represent the NULL type. Must be of length 1.
envir The environment in which to evaluate the arguments. If NULL, the environment associated with the object is used. If not available, the parent.frame() is used.

Value

A data.frame with each row containing the argument values of a method object.

See Also

Other lcMethod functions: [[,lcMethod-method, as.data.frame.lcMethod(), as.lcMethods(), as.list.lcMethod(), evaluate.lcMethod(), formula.lcMethod(), lcMethod-class, update.lcMethod()
as.lcMethods

Convert a list of lcMethod objects to a lcMethods list

Description
Convert a list of lcMethod objects to a lcMethods list

Usage
as.lcMethods(x)

Arguments
x
A list of lcMethod objects.

Value
A lcMethods object.

See Also
Other lcMethod functions: [[,lcMethod-method, as.data.frame.lcMethods(), as.data.frame.lcMethod(), as.list.lcMethod(), evaluate.lcMethod(), formula.lcMethod(), lcMethod-class, update.lcMethod()

as.lcModels

Convert a list of lcModels to a lcModels list

Description
Convert a list of lcModels to a lcModels list

Usage
as.lcModels(x)

Arguments
x
An R object.

Value
A lcModels object.

See Also
Other lcModel list functions: lcModels, print.lcModels(), subset.lcModels()
as.list.lcMethod

Extract the method arguments as a list

Description

Extract the method arguments as a list

Usage

## S3 method for class 'lcMethod'
as.list(x, ..., args = names(x), eval = TRUE, expand = FALSE, envir = NULL)

Arguments

x
  The lcMethod object.
...
  Additional arguments.
args
  A character vector of argument names to select. Only available arguments are
  returned. Alternatively, a function or list of functions, whose formal argu-
  ments will be selected from the method.
eval
  Whether to evaluate the arguments.
expand
  Whether to return all method arguments when "..." is present among the re-
  quested argument names.
eenvir
  The environment in which to evaluate the arguments. If NULL, the environment
  associated with the object is used. If not available, the parent.frame() is used.

Value

A list with the argument calls or evaluated results depending on the value for eval.

See Also

Other lcMethod functions: [[,lcMethod-method,as.data.frame.lcMethods(),as.data.frame.lcMethod(),
as.lcMethods(),evaluate.lcMethod(),formula.lcMethod(),lcMethod-class,update.lcMethod()]

Examples

data(latrendData)
method <- lcMethodKML("Y", id = "Id", time = "Time")
as.list(method)

as.list(method, args = c('id', 'time'))

# select arguments used by kml()
as.list(method, args = kml::kml)

# select arguments used by either kml() or parALGO()
as.list(method, args = c(kml::kml, kml::parALGO))
clusterNames<-

**clusterNames**  
*Get the cluster names*

**Description**
Get the cluster names

**Usage**
```r
clusterNames(object, factor = FALSE)
```

**Arguments**
- `object`  
The `lcModel` object.
- `factor`  
Whether to return the cluster names as a factor.

**Value**
A character of the cluster names.

**Examples**
```r
data(latrendData)
model <- latrend(lcMethodKML("Y", id = "Id", time = "Time"), latrendData)
clusterNames(model) # A, B
```

---

clusterNames<-

**clusterNames**  
*Update the cluster names*

**Description**
Update the cluster names

**Usage**
```r
clusterNames(object) <- value
```

**Arguments**
- `object`  
The `lcModel` object to update.
- `value`  
The character with the new names.

**Value**
The updated `lcModel` object.
Examples

```r
data(latrendData)
model <- latrend(lcMethodKML("Y", id = "Id", time = "Time"), latrendData)
clusterNames(model) <- c("Group 1", "Group 2")
```

clusterProportions

Proportional size of each cluster

Description

Proportional size of each cluster

Usage

```r
## S4 method for signature 'lcModel'
clusterProportions(object, ...)
```

Arguments

- **object**
  - The lcModel to obtain the proportions from.
- **...**
  - Not used.

Examples

```r
data(latrendData)
model <- latrend(lcMethodKML("Y", id = "Id", time = "Time"), latrendData)
clusterProportions(model)
```

clusterSizes

Number of strata per cluster

Description

Number of strata per cluster

Usage

```r
clusterSizes(object)
```

Arguments

- **object**
  - The lcModel object.

Examples

```r
model <- latrend(lcMethodKML("Y", id = "Id", time = "Time"), latrendData)
clusterSizes(model)
```
clusterTrajectories  

Extract the cluster trajectories

Description

Extracts a data frame of all cluster trajectories.

Usage

```r
## S4 method for signature 'lcModel'
clusterTrajectories(object, at = time(object), what = "mu", ...)
```

Arguments

- `object`: The `lcModel` object.
- `at`: An optional vector, list or data frame of covariates at which to compute the cluster trajectory predictions. If a vector is specified, this is assumed to be the time covariate. Otherwise, a named list or data frame must be provided.
- `what`: The distributional parameter to predict. By default, the mean response 'mu' is predicted. The cluster membership predictions can be obtained by specifying what='mb'.
- `...`: Additional arguments.

Value

A data.frame of the estimated values at the given times

See Also

Other model-specific methods: coef.lcModel(), converged(), deviance.lcModel(), df.residual.lcModel(), fitted.lcModel(), lcModel-class, logLik.lcModel(), model.frame.lcModel(), nobs.lcModel(), postprob(), predict.lcModel(), predictAssignments(), predictForCluster(), predictPostprob(), residuals.lcModel(), sigma.lcModel(), time.lcModel(), trajectories()

Examples

```r
model <- latrend(method = lcMethodLcmmGMM(Y ~ Time + (1 | Id)),
id = "Id", time = "Time", data = latrendData)
clusterTrajectories(model)

clusterTrajectories(model, at = c(0, .5, 1))
```
coef.lcModel

Coefficients of a lcModel

Description

Coefficients of a lcModel

Usage

```r
## S3 method for class 'lcModel'
coef(object, ...)
```

Arguments

- `object`: The lcModel object.
- `...`: Additional arguments.

Value

A named numeric vector with all coefficients, or a matrix with each column containing the cluster-specific coefficients.

See Also

Other model-specific methods: `clusterTrajectories()`, `converged()`, `deviance.lcModel()`, `df.residual.lcModel()`, `fitted.lcModel()`, `lcModel-class`, `logLik.lcModel()`, `model.frame.lcModel()`, `nobs.lcModel()`, `postprob()`, `predict.lcModel()`, `predictAssignments()`, `predictForCluster()`, `predictPostprob()`, `residuals.lcModel()`, `sigma.lcModel()`, `time.lcModel()`, `trajectories()`

confusionMatrix

Compute the posterior confusion matrix

Description

Compute a nClusters x nClusters posterior confusion matrix (PCM). The entry (i,j) represents the probability of a trajectory belonging to class i is assigned to class j under a given assignment strategy.

Usage

```r
confusionMatrix(object, strategy = which.max, scale = TRUE)
```
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>The object.</td>
</tr>
<tr>
<td>strategy</td>
<td>The <code>trajectoryAssignments</code> strategy to compute the PCM under. If <code>strategy = NULL</code>, weighted random assignment is assumed (analogous to a repeated <code>[which.weight]</code> strategy evaluation).</td>
</tr>
<tr>
<td>scale</td>
<td>Whether to express the confusion in probabilities (<code>scale = TRUE</code>), or in the number of trajectories.</td>
</tr>
</tbody>
</table>

Examples

```r
data(latrendData)
model = latrend(method=lcMethodLcmmGMM(Y ~ CLUSTER * Time + (1 | Id), id = "Id", time = "Time"),
data=latrendData)
confusionMatrix(model)
```

---

converged: Check model convergence

Description

Check convergence of the fitted model.

Usage

```r
## S4 method for signature 'lcModel'
converged(object, ...)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>The <code>lcModel</code> to check for convergence.</td>
</tr>
<tr>
<td>...</td>
<td>Additional arguments.</td>
</tr>
</tbody>
</table>

Value

Either logical indicating convergence, or a numeric status code.

See Also

Other model-specific methods: `clusterTrajectories()`, `coef.lcModel()`, `deviance.lcModel()`, `df.residual.lcModel()`, `Fitted.lcModel()`, `lcModel-class`, `logLik.lcModel()`, `model.frame.lcModel()`, `nobs.lcModel()`, `postprob()`, `predict.lcModel()`, `predictAssignments()`, `predictForCluster()`, `predictPostprob()`, `residuals.lcModel()`, `sigma.lcModel()`, `time.lcModel()`, `trajectories()`
createTestDataFold  
Create the test fold data for validation

Description
Create the test fold data for validation

Usage
createTestDataFold(data, trainData, id = getOption("latrend.id"))

Arguments
- data: A data.frame representing the complete dataset.
- trainData: A data.frame representing the training data, which should be a subset of data.
- id: The trajectory identifier variable.

See Also
createTrainDataFolds

Other validation methods: createTestDataFolds(), createTrainDataFolds(), latrendBoot(), latrendCV(), lcModel-data-filters

Examples

```r
data(latrendData)
trainDataList <- createTrainDataFolds(latrendData, id = "Id", folds = 10)
testData1 <- createTestDataFold(latrendData, trainDataList[[1]], id = "Id")
```

createTestDataFolds  
Create all k test folds from the training data

Description
Create all k test folds from the training data

Usage
createTestDataFolds(data, trainDataList, ...)

Arguments
- data: A data.frame representing the complete dataset.
- trainDataList: A list of data.frame representing each of the data training folds. These should be derived from data.
- ...: Arguments passed to createTestDataFold.
createTrainDataFolds

See Also

Other validation methods: `createTestDataFold()`, `createTrainDataFolds()`, `latrendBoot()`, `latrendCV()`, `lcModel-data-filters`

Examples

data(latrendData)
trainDataList <- createTrainDataFolds(latrendData, folds = 10, id = "Id")
testDataList <- createTestDataFolds(latrendData, trainDataList)

createTrainDataFolds

Create the training data for each of the k models in k-fold cross validation evaluation

Description

Create the training data for each of the k models in k-fold cross validation evaluation

Usage

```r
createTrainDataFolds(
  data,
  folds = 10,
  id = getOption("latrend.id"),
  seed = NULL
)
```

Arguments

- `data`: A `data.frame` representing the complete dataset.
- `folds`: The number of folds. By default, a 10-fold scheme is used.
- `id`: The trajectory identifier variable.
- `seed`: The seed to use, in order to ensure reproducible fold generation at a later moment.

Value

A list of `data.frame` of the folds training datasets.

See Also

Other validation methods: `createTestDataFold()`, `createTestDataFolds()`, `latrendBoot()`, `latrendCV()`, `lcModel-data-filters`
**Examples**

```r
data(latrendData)
trainFolds <- createTrainDataFolds(latrendData, folds = 10, id = "Id")

trainFolds <- createTrainDataFolds(latrendData, folds = 10, id = "Id", seed = 1)
```

**dcastRepeatedMeasures**  
*Cast a longitudinal data.frame to a matrix*

**Description**

Converts a longitudinal data.frame comprising trajectories with an equal number of observations, measured at identical moments in time, to a matrix. Each row of the matrix represents a trajectory.

**Usage**

```r
dcastRepeatedMeasures(data, response, id = getOption("latrend.id"), time = getOption("latrend.time")
```

**Arguments**

- **data**: The matrix containing a trajectory on each row.
- **response**: The response column name.
- **id**: The id column name.
- **time**: The time column name.

**Value**

A matrix with a trajectory per row.

---

**defineExternalMetric**  
*Define an external metric for lcModels*

**Description**

Define an external metric for lcModels

**Usage**

```r
defineExternalMetric(name, fun, warnIfExists = TRUE)
```
defineInternalMetric

Arguments

name The name of the metric.
fun The function to compute the metric, accepting a lcModel object as input.
warnIfExists Whether to output a warning when the new metric is already defined.

See Also

Other metric functions: `defineInternalMetric()`, `externalMetric`, `lcModel`, `lcModel-method`, `getExternalMetricDefinition()`, `getExternalMetricNames()`, `getInternalMetricDefinition()`, `getInternalMetricNames()`, `metric()`

---

defineInternalMetric Define an internal metric for lcModels

Description

Define an internal metric for lcModels

Usage

```r
defineInternalMetric(name, fun, warnIfExists = TRUE)
```

Arguments

name The name of the metric.
fun The function to compute the metric, accepting a lcModel object as input.
warnIfExists Whether to output a warning when the new metric is already defined.

See Also

Other metric functions: `defineExternalMetric()`, `externalMetric`, `lcModel`, `lcModel-method`, `getExternalMetricDefinition()`, `getExternalMetricNames()`, `getInternalMetricDefinition()`, `getInternalMetricNames()`, `metric()`
deviance.lcModel

## S3 method for class 'lcModel'
deviance(object, ...)

Arguments

object The lcModel object.

... Additional arguments.

See Also

Other model-specific methods: clusterTrajectories(), coef.lcModel(), converged(), df.residual.lcModel(), fitted.lcModel(), lcModel-class, logLik.lcModel(), model.frame.lcModel(), nobs.lcModel(), postprob(), predict.lcModel(), predictAssignments(), predictForCluster(), predictPostprob(), residuals.lcModel(), sigma.lcModel(), time.lcModel(), trajectories()

df.residual.lcModel

Extract the residual degrees of freedom from a lcModel

## S3 method for class 'lcModel'
df.residual(object, ...)

Arguments

object The lcModel object.

... Additional arguments.

See Also

Other model-specific methods: clusterTrajectories(), coef.lcModel(), converged(), deviance.lcModel(), fitted.lcModel(), lcModel-class, logLik.lcModel(), model.frame.lcModel(), nobs.lcModel(), postprob(), predict.lcModel(), predictAssignments(), predictForCluster(), predictPostprob(), residuals.lcModel(), sigma.lcModel(), time.lcModel(), trajectories()
estimationTime

Get the model estimation time

Description
Get the model estimation time

Usage
estimationTime(object)

Arguments
object The lcModel object.

Value
The model estimation time in seconds.

evaluate.lcMethod

Substitute the call arguments for their evaluated values

Description
Substitutes the call arguments if they can be evaluated without error.

Usage
## S3 method for class 'lcMethod'
evaluate(
  object,
  classes = "ANY",
  try = TRUE,
  exclude = character(),
  envir = NULL
)

Arguments
object The lcMethod object.

object

classes Substituted only arguments with specific class types. By default, all types are

try Whether to try to evaluate arguments and ignore errors (the default), or to fail

exclude Arguments to exclude from evaluation.

on any argument evaluation error.

envir The environment in which to evaluate the arguments. If NULL, the environment

associated with the object is used. If not available, the parent.frame() is used.
Value

A new lcMethod object with the substituted arguments.

See Also

Other lcMethod functions: 
\[\text{[}, \text{lcMethod-method}, \text{as.data.frame.lcMethods()}, \text{as.data.frame.lcMethod()}, \text{as.lcMethods()}, \text{as.list.lcMethod()}, \text{formula.lcMethod()}, \text{lcMethod-class}, \text{update.lcMethod()}\]

---

### Description

Compute external model metric(s)

### Usage

```r
## S4 method for signature 'lcModel,lcModel'
externalMetric(object, object2, name, ...)

## S4 method for signature 'lcModels,missing'
externalMetric(object, object2, name = "adjustedRand")

## S4 method for signature 'lcModels,lcModel'
externalMetric(object, object2, name, drop = TRUE)

## S4 method for signature 'list,lcModel'
externalMetric(object, object2, name, drop = TRUE)
```

### Arguments

- `object`: The lcModel, lcModels, or list of lcModel objects to compute the metrics for.
- `object2`: The other lcModel to compare with.
- `name`: The name(s) of the metric(s) to compute.
- `...`: Additional arguments.
- `drop`: Whether to drop the matrix dimensions in case of a single model output.

### Value

For `externalMetric(lcModel,lcModel)`: A numeric vector of the computed metrics. A named numeric vector containing the computed model metrics.

For `externalMetric(lcModels)`: A distance matrix of class `dist` representing the pairwise comparisons.
For `externalMetric(lcModels,lcModel)`: A named numeric vector or matrix containing the computed model metrics.

For `externalMetric(list,lcModel)`: A named numeric vector or matrix containing the computed model metrics.

See Also

- `metric`
- Other metric functions: `defineExternalMetric()`, `defineInternalMetric()`, `getExternalMetricDefinition()`, `getExternalMetricNames()`, `getInternalMetricDefinition()`, `getInternalMetricNames()`.

Examples

```r
    data(latrendData)
    model1 <- latrend(lcMethodKML("Y", id = "Id", time = "Time"), latrendData)
    model2 <- latrend(lcMethodLcmmGMM(Y ~ Time + (1 | Id), id = "Id", time = "Time"), latrendData)
    ari <- externalMetric(model1, model2, "adjustedRand")
```

---

**fitted.lcModel**

*Extract lcModel fitted values*

**Description**

Extract lcModel fitted values

**Usage**

```r
## S3 method for class 'lcModel'
fitted(object, ..., clusters = trajectoryAssignments(object))
```

**Arguments**

- **object**: The lcModel object.
- **...**: Additional arguments.
- **clusters**: Optional cluster assignments per id. If unspecified, a matrix is returned containing the cluster-specific predictions per column.

**Value**

A numeric vector of the fitted values for the respective class, or a matrix of fitted values for each cluster.

**See Also**

Other model-specific methods: `clusterTrajectories()`, `coef.lcModel()`, `converged()`, `deviance.lcModel()`, `df.residual.lcModel()`, `lcModel-class`, `logLik.lcModel()`, `model.frame.lcModel()`, `nobs.lcModel()`, `postprob()`, `predict.lcModel()`, `predictAssignments()`, `predictForCluster()`, `predictPostprob()`, `residuals.lcModel()`, `sigma.lcModel()`, `time.lcModel()`, `trajectories()`
**formula.lcMethod**

Extract formula

---

**Description**

Extracts the associated formula for the given distributional parameter.

**Usage**

```r
## S3 method for class 'lcMethod'
formula(x, what = "mu", envir = NULL, ...)
```

**Arguments**

- `x` The lcMethod object.
- `what` The distributional parameter to which this formula applies. By default, the formula specifies "mu".
- `envir` The environment in which to evaluate the arguments. If `NULL`, the environment associated with the object is used. If not available, the `parent.frame()` is used.
- `...` Additional arguments.

**Value**

The formula for the given distributional parameter.

**See Also**

Other lcMethod functions: `.lcMethod-method, as.data.frame.lcMethods(), as.data.frame.lcMethod(), as.lcMethods(), as.list.lcMethod(), evaluate.lcMethod(), lcMethod-class, update.lcMethod()

**Examples**

```r
m <- lcMethodLcmGBTM(Value ~ Time)
formula(m) # Value ~ Time
formula(m, what = "mb") # ~1
```
formula.lcModel  Extract the formula of a lcModel

Description
Extract the formula of a lcModel

Usage
```r
## S3 method for class 'lcModel'
formula(x, what = "mu", ...)
```

Arguments
- `x`: The lcModel object.
- `what`: The distributional parameter
- `...`: Additional arguments.

Value
Returns the associated formula, or \(~0\) if not specified.

generateLongData  Generate longitudinal test data

Description
Generate longitudinal test data

Usage
```r
generateLongData(
  sizes = c(40, 60),
  fixed = Value ~ 1 + Time,
  cluster = ~1 + Time,
  random = ~1,
  id =getOption("latrend.id"),
  data = data.frame(Time = seq(0, 1, by = 0.1)),
  fixedCoefs = c(0, 0),
  clusterCoefs = cbind(c(-2, 1), c(2, -1)),
  randomScales = cbind(0.1, 0.1),
  rrandom = rnorm,
  noiseScales = c(0.1, 0.1),
  rnoise = rnorm,
  clusterNames = LETTERS[seq_along(sizes)],
  shuffle = FALSE
)
```
getExternalMetricDefinition

Arguments

- **sizes**: Number of strata per cluster.
- **fixed**: Fixed effects formula.
- **cluster**: Cluster effects formula.
- **random**: Random effects formula.
- **id**: Name of the strata.
- **data**: Data with covariates to use for generation. Stratified data may be specified by adding a grouping column.
- **fixedCoefs**: Coefficients matrix for the fixed effects.
- **clusterCoefs**: Coefficients matrix for the cluster effects.
- **randomScales**: Standard deviations matrix for the size of the variance components (random effects).
- **rrandom**: Random sampler for generating the variance components at location 0.
- **noiseScales**: Scale of the random noise passed to rnoise. Either scalar or defined per cluster.
- **rnoise**: Random sampler for generating noise at location 0 with the respective scale.
- **clusterNames**: A character vector denoting the names of the generated clusters.
- **shuffle**: Whether to randomly reorder the strata in which they appear in the data.frame.

Examples

```r
longdata <- generateLongData(sizes = c(40, 70), id = "Id",
                             cluster = ~poly(Time, 2, raw = TRUE),
                             clusterCoefs = cbind(c(1, 2, 5), c(-3, 4, .2)))
plotTrajectories(longdata, response = "Value", id = "Id", time = "Time")
```

getExternalMetricDefinition

*Get the external metric definition*

Description

Get the external metric definition

Usage

getExternalMetricDefinition(name)

Arguments

- **name**: The name of the metric.

Value

The metric function, or NULL if not defined.
getInternalMetricDefinition

See Also

Other metric functions: `defineExternalMetric()`, `defineInternalMetric()`, `externalMetric`, `lcModel`, `lcModel-method`, `getExternalMetricNames()`, `getInternalMetricDefinition()`, `getInternalMetricNames()`, `metric()`

---

getExternalMetricNames

*Get the names of the available external metrics*

Description

Get the names of the available external metrics

Usage

getExternalMetricNames()

See Also

Other metric functions: `defineExternalMetric()`, `defineInternalMetric()`, `externalMetric`, `lcModel`, `lcModel-method`, `getExternalMetricDefinition()`, `getInternalMetricDefinition()`, `getInternalMetricNames()`, `metric()`

---

getInternalMetricDefinition

*Get the internal metric definition*

Description

Get the internal metric definition

Usage

getInternalMetricDefinition(name)

Arguments

name

The name of the metric.

Value

The metric function, or NULL if not defined.

See Also

Other metric functions: `defineExternalMetric()`, `defineInternalMetric()`, `externalMetric`, `lcModel`, `lcModel-method`, `getExternalMetricDefinition()`, `getExternalMetricNames()`, `getInternalMetricNames()`, `metric()`
getInternalMetricNames

*Get the names of the available internal metrics*

**Description**

Get the names of the available internal metrics

**Usage**

```r
getInternalMetricNames()
```

**See Also**

Other metric functions: `defineExternalMetric()`, `defineInternalMetric()`, `externalMetric`, `lcModel`, `lcModel-method`, `getExternalMetricDefinition()`, `getExternalMetricNames()`, `getInternalMetricDefinition()`, `metric()`

---

getLcMethod

*Get the method specification of a lcModel*

**Description**

Get the method specification of a lcModel

**Usage**

```r
getCodeMethod(object)
```

**Arguments**

- `object` The `lcModel` object.

**Examples**

```r
model = latrend(method=lcMethodKML("Y", id = "Id", time = "Time"), data=latrendData)
getCodeMethod(model)
```
ids

Get the unique ids included in this model

Description
Get the unique ids included in this model

Usage
ids(object)

Arguments
object The lcModel object.

Details
The order returned by ids(lcModel) determines the id order for any output involving id-specific values, such as in trajectoryAssignments() or postprob()

Examples
model = latrend(lcMethodKML("Y", id = "Id", time = "Time"), latrendData)
ids(model) # S1, S2, ..., S500

idVariable

Extract the trajectory identifier variable

Description
Extracts the trajectory identifier variable (i.e., column name) from the given object.

Usage
## S4 method for signature 'lcMethod'
idVariable(object, ...)

## S4 method for signature 'lcModel'
idVariable(object)

Arguments
object The object to extract the variable from.
... Not used.
Value

The trajectory identifier name, as character.

See Also

Other lcModel variables: `responseVariable()`, `timeVariable()`

Examples

```r
method <- lcMethodKML(id = "Traj")
idVariable(method) # "Traj"

model <- latrend(lcMethodKML("Y", id = "Id", time = "Time"), latrendData)
idVariable(model) # "Id"
```

Description

Cluster longitudinal data

Usage

```r
latrend(
  method,       # The lcMethod object specifying the longitudinal cluster method to apply.
  data,         # The data.frame or matrix to which to apply the method.
  ...,          # Any other arguments to update the lcMethod definition with.
  envir = NULL, # The environment in which to evaluate the method arguments. Note that this
  verbose = getOption("latrend.verbose")
)
```

Arguments

- `method` The lcMethod object specifying the longitudinal cluster method to apply.
- `data` The data.frame or matrix to which to apply the method.
- `...` Any other arguments to update the lcMethod definition with.
- `envir` The environment in which to evaluate the method arguments. Note that this only applies to data when data is a call.
- `verbose` The level of verbosity. Either an object of class Verbose (see `R.utils::Verbose` for details), a logical indicating whether to show basic computation information, a numeric indicating the verbosity level (see `Verbose`), or one of c(‘info’, ‘fine’, ‘finest’).

Details

If a seed value is specified in the lcMethod object or arguments to `latrend`, this seed is set using `set.seed` prior to the cluster preparation step.
latrendBatch

Value

A lcModel object representing the fitted model.

See Also

Other longitudinal cluster fit functions: latrendBatch(), latrendBoot(), latrendCV(), latrendRep()

Examples

data(latrendData)
model <- latrend(lcMethodKML("Y", id = "Id", time = "Time"), data = latrendData)

method <- lcMethodKML("Y", id = "Id", time = "Time")
model <- latrend(method, data = latrendData, nClusters = 3)

model <- latrend(method, data = latrendData, nClusters = 3, seed = 1)

latrendBatchCluster longitudinal data for a list of model specifications

Description

Fit a list of longitudinal cluster methods.

Usage

latrendBatch(
  methods,
  data,
  cartesian = TRUE,
  envir = NULL,
  verbose = getOption("latrend.verbose")
)

Arguments

methods A list of lcMethod objects.
data A data.frame, matrix, or a list thereof to which to apply the respective lcMethod. Multiple datasets can be supplied by encapsulating the datasets using data=.(df1,df2,...,dfN).
cartesian Whether to fit the provided methods on each of the datasets. If cartesian=FALSE, only a single dataset may be provided or a list of data matching the length of methods.
envir The environment in which to evaluate the lcMethod arguments.
verbose The level of verbosity. Either an object of class Verbose (see R.utils::Verbose for details), a logical indicating whether to show basic computation information, a numeric indicating the verbosity level (see Verbose), or one of c('info','fine','finest').
latrendBoot

Value

A lcModels object.

See Also

lcMethods
Other longitudinal cluster fit functions: latrendBoot(), latrendCV(), latrendRep(), latrend()

Examples

data(latrendData)
methods <- lcMethods(lcMethodKML("Y", id = "Id", time = "Time"), nClusters = 1:3)
models <- latrendBatch(methods, data = latrendData)

models <- latrendBatch(lcMethods(lcMethodKML("Y", id = "Id", time = "Time"), nClusters = 1:2),
  data = .(subset(latrendData, Time > .5),
    subset(latrendData, Time < .5))) # different data per method

latrendBoot

Cluster longitudinal data using bootstrapping

Description

Performs bootstrapping, generating samples from the given data at the id level, fitting a lcModel to each sample.

Usage

latrendBoot(
  method,
  data,
  samples = 50,
  seed = NULL,
  envir = NULL,
  verbose =getOption("latrend.verbose")
)

Arguments

method The lcMethod object specifying the longitudinal cluster method to apply.
data A data.frame.
samples The number of bootstrap samples to evaluate.
seed The seed to use. Optional.
envir The environment in which to evaluate the method arguments. Note that this only applies to data when data is a call.
verbose The level of verbosity. Either an object of class `Verbose` (see `R.utils::Verbose` for details), a logical indicating whether to show basic computation information, a numeric indicating the verbosity level (see `Verbose`), or one of c('info', 'fine', 'finest').

Value
A `lcModels` object of length `samples`.

See Also
Other longitudinal cluster fit functions: `latrendBatch()`, `latrendCV()`, `latrendRep()`, `latrend()`
Other validation methods: `createTestDataFolds()`, `createTestDataFold()`, `createTrainDataFolds()`, `latrendCV()`, `lcModel-data-filters`

Examples
```r
data(latrendData)
method <- lcMethodKML("Y", id = "Id", time = "Time")
model <- latrendBoot(method, latrendData, samples = 10)
```

Description
Apply k-fold cross validation for internal cluster validation. Creates k random subsets ("folds") from the data, estimating a model for each of the k-1 combined folds.

Usage
```r
latrendCV(
  method,  # The lcMethod object specifying the longitudinal cluster method to apply.
  data,    # A data.frame.
  folds = 10,  # The number of folds. Ten folds by default.
  seed = NULL,  # The seed to use. Optional.
  envir = NULL,  # The environment in which to evaluate the method arguments. Note that this only applies to data when data is a call.
  verbose = getOption("latrend.verbose")
)
```

Arguments
- `method` The lcMethod object specifying the longitudinal cluster method to apply.
- `data` A data.frame.
- `folds` The number of folds. Ten folds by default.
- `seed` The seed to use. Optional.
- `envir` The environment in which to evaluate the method arguments. Note that this only applies to data when data is a call.
verbose
The level of verbosity. Either an object of class Verbose (see \texttt{R.utils::Verbose}
for details), a logical indicating whether to show basic computation information,
a numeric indicating the verbosity level (see \texttt{Verbose}), or one of \texttt{c('info', 'fine', 'finest')}.

Value
A \texttt{lcModels} object of containing the folds training models.

See Also
Other longitudinal cluster fit functions: \texttt{latrendBatch()}, \texttt{latrendBoot()}, \texttt{latrendRep()}, \texttt{latrend()}
Other validation methods: \texttt{createTestDataFolds()}, \texttt{createTestDataFold()}, \texttt{createTrainDataFolds()},
\texttt{latrendBoot()}, \texttt{lcModel-data-filters}

Examples
\begin{verbatim}
data(latrendData)
method <- lcMethodKML("Y", id = "Id", time = "Time")
model <- latrendCV(method, latrendData, folds = 5)

model <- latrendCV(method, subset(latrendData, Time < .5), folds = 5, seed = 1)
\end{verbatim}

Description
Synthetic longitudinal dataset comprising three classes

Usage
\texttt{latrendData}

Format
A \texttt{data.frame} describing 200 trajectories originating from one of three classes, each
with a different cluster trajectory. Trajectories randomly deviate in intercept and slope from the reference
cluster.

\begin{description}
\item[Id] trajectory identifier, integer.
\item[Time] measurement time, numeric between 0 and 2.
\item[Y] observed variable, numeric.
\item[Class] the reference class, factor.
\end{description}

Source
This dataset was generated using \texttt{generateLongData}.
latrendRep

Cluster longitudinal data repeatedly

Description

Performs a repeated fit of the specified latrend model on the given data.

Usage

latrendRep(
method,
data,
.rep = 10,
..., .errorhandling = "remove",
.seed = NULL,
envir = NULL,
verbose =getOption("latrend.verbose")
)

Arguments

method  The lcMethod object specifying the longitudinal cluster method to apply.
data  The data.frame or matrix to which to apply the method.
.rep  The number of repeated fits.
...  Any other arguments to update the lcMethod definition with.
.errorhandling  How to handle fits in which on error occurs. If "remove", errors are ignored and the respective repetition is exempt from the returned model list. If "stop", errors are not caught, ensuring that the function halts on the first error.
.seed  Set the seed for generating the respective seed for each of the repeated fits.
envir  The environment in which to evaluate the method arguments. Note that this only applies to data when data is a call.
.verbose  The level of verbosity. Either an object of class Verbose (see R.utils::Verbose for details), a logical indicating whether to show basic computation information, a numeric indicating the verbosity level (see Verbose), or one of c('info', 'fine', 'finest').

Details

This method is faster than repeatedly calling latrend as it only prepares the data via prepareData() once.

See Also

generateLongData
lcApproxModel-class

Value

A lcModels object containing the resulting models.

See Also

Other longitudinal cluster fit functions: latrendBatch(), latrendBoot(), latrendCV(), latrend()

Examples

data(latrendData)
method <- lcMethodKML("Y", id = "Id", time = "Time")
models <- latrendRep(method, data = latrendData, .rep = 5) # 5 repeated runs

models <- latrendRep(method, data = latrendData, .seed = 1, .rep = 3)

Description

approx models have defined cluster trajectories at fixed moments in time, which should be interpolated. For a correct implementation, lcApproxModel requires the extending class to implement clusterTrajectories(at=NULL) to return the fixed cluster trajectories.

Usage

## S3 method for class 'lcApproxModel'
fitted(object, ..., clusters = trajectoryAssignments(object))

## S3 method for class 'lcApproxModel'
predict(object, ..., newdata = NULL, what = "mu", approxFun = approx)

Arguments

object
  The lcModel object.

...  Additional arguments.

clusters
  Optional cluster assignments per id. If unspecified, a matrix is returned containing the cluster-specific predictions per column.

newdata
  Optional data frame for which to compute the model predictions. If omitted, the model training data is used. Cluster trajectory predictions are made when ids are not specified. If the clusters are specified under the Cluster column, output is given only for the specified cluster. Otherwise, a matrix is returned with predictions for all clusters.

what
  The distributional parameter to predict. By default, the mean response 'mu' is predicted. The cluster membership predictions can be obtained by specifying what='mb'.

approxFun
  The interpolation function to use for time points not in the feature set.
lcMethod-class

Create a lcMethod object of the specified type and arguments

Description

Provides a mechanism for creating lcMethod objects for an arbitrary class. Note that it is advisable to use the class-specific constructors instead.

Usage

lcMethod(.class, ..., .defaults = list(), .excludeArgs = c())

Arguments

Argument | Description
--- | ---
.class | The type of lcMethod-class class
... | Any arguments to assign to the method object.
.defaults | See defaults of lcMethod.call.
.excludeArgs | See excludeArgs of lcMethod.call.

See Also

lcMethod.call

Description

Base class used to define a longitudinal cluster method. It is implemented as a wrapper around a call.

Model estimation is handled through a series of calls implement by the lcMethod object. The calls are made by latrend, in the following order:

- compose
- validate
- prepareData
- preFit
- fit
- postFit

Extracts the assigned label.

Extracts the name of the given object.
Usage

## S4 method for signature 'lcMethod'
compose(method, envir = NULL)

## S4 method for signature 'lcMethod'
fit(method, data, envir, verbose)

## S4 method for signature 'lcMethod'
getLabel(object, ...)

## S4 method for signature 'lcMethod'
getName(object)

## S4 method for signature 'lcMethod'
getShortName(object, ...)

## S4 method for signature 'lcMethod'
length(x)

## S4 method for signature 'lcMethod'
names(x)

## S4 method for signature 'lcMethod'
prefit(method, data, envir, verbose)

## S4 method for signature 'lcMethod'
prepareData(method, data, verbose)

## S4 method for signature 'lcMethod'
potify(method, data, model, envir, verbose)

## S4 method for signature 'lcMethod'
validate(method, data, envir = NULL, ...)

Arguments

- **method**: The `lcMethod` object.
- **envir**: The environment in which the `lcMethod` should be evaluated.
- **data**: The data, as a `data.frame`, on which the model will be trained.
- **verbose**: A `R.utils::Verbose` object indicating the level of verbosity.
- **object**: The object to extract the label from.
- **...**: Additional arguments.
- **x**: The `lcMethod` object.
- **model**: The `lcModel` object returned by `fit()`.
Details

Because the lcMethod arguments may be unevaluated, evaluation functions such as [[ accept an envir argument. A default environment can be assigned or obtained from a lcMethod object using the environment() function.

Value

The updated lcMethod object.

An lcModel object.

The extracted label, as character.

A character vector of argument names.

An environment that will be passed to fit().

A data.frame with the post-processed data.

The updated lcModel object.

Either TRUE if all validation checks passed, or a character containing a description of the failed validation checks.

Slots

arguments A list representing the arguments of the lcMethod object. Arguments are not evaluated upon creation of the method object. Instead, arguments are stored similar to a call object. Do not modify or access.

sourceCalls A list of calls for tracking the original call after substitution. Used for printing objects which require too many characters (e.g., function definitions, matrices).

See Also

environment


Other lcMethod functions: [[, lcMethod-method, as.data.frame.lcMethods(), as.data.frame.lcMethod(), as.lcMethods(), as.list.lcMethod(), evaluate.lcMethod(), formula.lcMethod(), update.lcMethod()

Examples

getName(lcMethodKML("Y")) # "longitudinal k-means"
getShortName(lcMethodKML("Y")) # "KML"
m = lcMethodKML("Y")
names(m)
lcMethod.call

Create a lcMethod object from a call

Description

Creates a lcMethod class of the specified type Class for the given arguments given in a call, along with any default arguments from reference functions. This function is intended to be used by classes extending lcMethod to provide an easy way to construct the appropriate call object.

Usage

lcMethod.call(Class, call, defaults = list(), excludeArgs = c())

Arguments

Class
The type of lcMethod class

call
The arguments to create the lcMethod from.

defaults
List of function to obtain defaults from for arguments not defined in call.

excludeArgs
The names of the arguments to exclude from the defaults, provided as a character vector.

Value

An object of class Class that extends lcMethod.

See Also

lcMethod

Examples

data(latrendData)
lcMethodKML2 <- function(response = "Y", id = "Id", time = "Time", nClusters = 2, ...)
{
lcMethod.call("lcMethodKML", call = stackoverflow::match.call.defaults(),
defaults = c(kml::kml, kml::parALGO),
excludeArgs = c("object", "nbClusters", "parAlgo", "toPlot", "saveFreq"))
}
method <- lcMethodKML2(nClusters = 3)
latrend(method, data = latrendData)
lcMethodAKMedoids  

Specify AKMedoids method

Description

Specify AKMedoids method

Usage

```
lcMethodAKMedoids(
  response,  
  time = getOption("latrend.time"), 
  id = getOption("latrend.id"), 
  nClusters = 3, 
  clusterCenter = median, 
  ...
)
```

Arguments

- **response**: The name of the response variable.
- **time**: The name of the time variable.
- **id**: The name of the trajectory identification variable.
- **nClusters**: The number of clusters to estimate.
- **clusterCenter**: A function for computing the cluster center representation.
- **...**: Arguments passed to `akmedoids::akmedoids.clust`. The following external arguments are ignored: traj, id_field, k

See Also

Other lcMethod implementations: 
- lcMethod-class, lcMethodCrimCV, lcMethodCustom, lcMethodDtwwclust, 
- lcMethodFunFEM, lcMethodGCKM, lcMethodKML, lcMethodLMKM, lcMethodLcmmGBTM, lcMethodLcmmGMM, 
- lcMethodLongclust, lcMethodMclustLLPA, lcMethodMixAK_GLMM, lcMethodMixtoolsGMM, lcMethodMixtoolsNPRM, 
- lcMethodRandom, lcMethodStratify, lcMethodTwoStep

Examples

```
library(akmedoids)
data(latrendData)
method <- lcMethodAKMedoids(response = "Y", time = "Time", id = "Id", nClusters = 3)
model <- latrend(model, data = latrendData)
```
Specify a zero-inflated repeated-measures GBTM method

Usage

```r
lcMethodCrimCV(
  response,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  nClusters = 2,
  ...
)
```

Arguments

- **response**: The name of the response variable.
- **time**: The name of the time variable.
- **id**: The name of the trajectory identifier variable.
- **nClusters**: The number of clusters to estimate.
- **...**: Arguments passed to `crimCV::crimCV`. The following external arguments are ignored: `Dat`, `ng`.

See Also


Examples

```r
library(crimCV)
data(latrendData)
method <- lcMethodCrimCV("Y", id = "Id", time = "Time", nClusters = 3, dpolyp = 1, init = 2)
model <- latrend(method, data = subset(latrendData, Time > .5))
plot(model)
data(TO1adj)
method <- lcMethodCrimCV(response = "Offenses", time = "Offense", id = "Subject",
  nClusters = 2, dpolyp = 1, init = 2)
model <- latrend(method, data = TO1adj[1:100, ])
```
lcMethodCustom  Specify a custom method based on a model function

Description

Specify a custom method based on a model function

Usage

lcMethodCustom(
  response,
  fun,
  center = meanNA,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  name = "custom"
)

Arguments

  response  The name of the response variable.
  fun       The cluster function with signature (method, data).
  center    Optional function for computing the longitudinal cluster centers, with signature (x).
  time      The name of the time variable.
  id        The name of the trajectory identification variable.
  name      The name of the method.

See Also


Examples

data(latrendData)
# Stratification based on the mean response level
clusfun <- function(data, response, id, time, ...) {
  clusters <- data.table::as.data.table(data)[, mean(Y) > 0, by = Id]$V1
  lcModelCustom(data = data,
    trajectoryAssignments = factor(clusters, levels = c(FALSE, TRUE), labels = c("Low", "High")),
    response = response,
    time = time,
    id = id)
}
lcMethodDtwclust

method <- lcMethodCustom(response = "Y", fun = clusfun, id = "Id", time = "Time")
model <- latrend(method, data = latrendData)

--

lcMethodDtwclust  Specify time series clustering via dtwclust

Description

Specify time series clustering via dtwclust

Usage

lcMethodDtwclust(
  response,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  nClusters = 2,
  ...
)

Arguments

  response          The name of the response variable.
  time              The name of the time variable.
  id                The name of the trajectory identifier variable.
  nClusters         Number of clusters.
  ...               Arguments passed to dtwclust::tsclust. The following arguments are ignored: series, k, trace.

See Also


Examples

library(dtwclust)
data(latrendData)
method <- lcMethodDtwclust("Y", id = "Id", time = "Time", nClusters = 3)
model <- latrend(method, latrendData)
lcMethodFlexmix  

Method interface to flexmix()

Description

Wrapper to the flexmix() method from the flexmix package.

Usage

```r
lcMethodFlexmix(
  formula,
  formula.mb = ~1,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  nClusters = 2,
  ...
)
```

Arguments

- **formula**: A formula specifying the model.
- **formula.mb**: A formula specifying the class membership model. By default, an intercept-only model is used.
- **time**: The name of the time variable.
- **id**: The name of the trajectory identifier variable.
- **nClusters**: The number of clusters to estimate.
- **...**: Arguments passed to `flexmix::flexmix`. The following arguments are ignored: data, concomitant, k.

See Also

Other lcMethod package interfaces: `lcMethodFlexmixGBTM`

Examples

```r
library(flexmix)
data(latrendData)
method <- lcMethodFlexmix(Y ~ Time, id = "Id", time = "Time", nClusters = 3)
model <- latrend(method, latrendData)
```
Description

Fits a GBTM based on the `flexmix::FLXMRglm` driver.

Usage

```r
lcMethodFlexmixGBTM(
  formula,
  formula.mb = ~1,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  nClusters = 2,
  ...
)
```

Arguments

- `formula` A formula specifying the model.
- `formula.mb` A formula specifying the class membership model. By default, an intercept-only model is used.
- `time` The name of the time variable.
- `id` The name of the trajectory identifier variable.
- `nClusters` The number of clusters to estimate.
- `...` Arguments passed to `flexmix::flexmix` or `flexmix::FLXMRglm`. The following arguments are ignored: `data`, `k`, `trace`.

See Also

Other `lcMethod` package interfaces: `lcMethodFlexmix`

Examples

```r
library(flexmix)
data(latrendData)
method <- lcMethodFlexmixGBTM(Y ~ Time, id = "Id", time = "Time", nClusters = 3)
model <- latrend(method, latrendData)
```
lcMethodFunFEM

Specify a FunFEM method

Description

Specify a FunFEM method

Usage

lcMethodFunFEM(
response,
time = getOption("latrend.time"),
id = getOption("latrend.id"),
nClusters = 2,
basis = function(time) fda::create.bspline.basis(time, nbasis = 10, norder = 4),
...
)

Arguments

 response The name of the response variable.
 time The name of the time variable.
 id The name of the trajectory identifier variable.
 nClusters The number of clusters to estimate.
 basis The basis function. By default, a 3rd-order B-spline with 10 breaks is used.
 ... Arguments passed to funFEM::funFEM. The following external arguments are ignored: fd, K, disp, graph.

See Also


Examples

library(funFEM)
library(fda)
data(latrendData)
method <- lcMethodFunFEM("Y", id = "Id", time = "Time", nClusters = 3)
model <- latrend(method, latrendData)

method <- lcMethodFunFEM("Y",
basis = function(time) {
    create.bspline.basis(time,
        nbasis = 10, norder = 4)
})
Description

Two-step clustering through linear mixed modeling and k-means.

Usage

lcMethodGCKM(
  formula,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  nClusters = 2,
  center = meanNA,
  ...
)

Arguments

- **formula**: Formula, including a random effects component for the trajectory. See `lme4::lmer` formula syntax.
- **time**: The name of the time variable.
- **id**: The name of the trajectory identifier variable.
- **nClusters**: The number of clusters.
- **center**: Optional function for computing the longitudinal cluster centers, with signature \( \langle x \rangle \).
- **...**: Arguments passed to `lme4::lmer`. The following external arguments are ignored: data, centers, trace.

See Also


Examples

```r
library(lme4)
data(latrendData)
method <- lcMethodGCKM(Y ~ (Time | Id), id = "Id", time = "Time", nClusters = 3)
model <- latrend(method, latrendData)
```
lcMethodKML  

Specify a longitudinal k-means (KML) method

Description

Specify a longitudinal k-means (KML) method

Usage

```r
lcMethodKML(
  response,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  nClusters = 2,
  ...
)
```

Arguments

- **response**: The name of the response variable.
- **time**: The name of the time variable.
- **id**: The name of the trajectory identifier variable.
- **nClusters**: The number of clusters to estimate.
- **...**: Arguments passed to `kml::parALGO` and `kml::kml`. The following external arguments are ignored: object, nbClusters, parAlgo, toPlot, saveFreq

See Also


Examples

```r
library(kml)
data(latrendData)
method <- lcMethodKML("Y", id = "Id", time = "Time", nClusters = 3)
model <- latrend(method, latrendData)
```
lcMethodLcmmGBTM

Specify GBTM method

Description

Group-based trajectory modeling through fixed-effects modeling.

Usage

```r
lcMethodLcmmGBTM(
  formula,
  formula.mb = ~1,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  nClusters = 2,
  ...
)
```

Arguments

- **formula** A formula of the form `Response ~ Var1 + CLUSTER * Var2 + .`. Variables specified in the model are included as fixed effects. If an interaction is specified with the CLUSTER term then these covariates are included as fixed and mixture effects.
- **formula.mb** A formula specifying the class membership model. By default, an intercept-only model is used. This is a replacement of the internal classmb argument in `lcmm::hlme`.
- **time** The name of the time variable.
- **id** The name of the trajectory identifier variable. This replaces the subject argument of `lcmm::hlme`.
- **nClusters** The number of clusters to fit. This replaces the ng argument of `lcmm::hlme`.
- **...** Arguments passed to `lcmm::hlme`. The following arguments are ignored: data, fixed, random, mixture, subject, classmb, returndata, ng, verbose, subset.

See Also


Examples

```r
data(latrendData)
method <- lcMethodLcmmGBTM(Y ~ CLUSTER, id = "Id", time = "Time", nClusters = 3)
gbtm <- latrend(method, data = latrendData)
summary(gbtm)
```
lcMethodLcmmGMM

Specify GMM method using lcmm

Description

Growth mixture modeling through latent-class linear mixed modeling.

Usage

lcMethodLcmmGMM(
  formula,
  formula.mb = ~1,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  nClusters = 2,
  ...
)

Arguments

- **formula**: A formula of the form Response ~ Var1 + CLUSTER * Var2 + . + (Random1 + Random2 + . | Id). Variables specified in the model are included as fixed effects. If an interaction is specified with the CLUSTER term, then these covariates are included as fixed and mixture effects. The formula must contain a single random-effects component of the form (. | Id), where Id matches the name specified in the id argument, or ID (which will be substituted by the id argument). The random effects are cluster-specific.
- **formula.mb**: A formula specifying the class membership model. By default, an intercept-only model is used. This is a replacement of the internal classmb argument in lcmm::hlme.
- **time**: The name of the time variable.
- **id**: The name of the trajectory identifier variable. This replaces the subject argument of lcmm::hlme.
- **nClusters**: The number of clusters to fit. This replaces the ng argument of lcmm::hlme.
- **...**: Arguments passed to lcmm::hlme. The following arguments are ignored: data, fixed, random, mixture, subject, classmb, returndata, ng, verbose, subset.

Details

The formula argument is used to generate the fixed, random, and mixture arguments for lcmm::hlme.
See Also


Examples

data(latrendData)
method <- lcMethodLcmmGMM(Y ~ Time * CLUSTER + (1 | Id),
                           id = "Id", time = "Time", nClusters = 3)
gmm <- latrend(method, data = latrendData)
summary(gmm)

method <- lcMethodLcmmGMM(Y ~ Time * CLUSTER + (Time | Id),
                           id = "Id", time = "Time", nClusters = 3)

lcMethodLMKM

Two-step clustering through linear regression modeling and k-means

Description

Two-step clustering through linear regression modeling and k-means

Usage

lcMethodLMKM(
  formula,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  nClusters = 2,
  standardize = scale,
  ...
)

Arguments

formula A formula specifying the linear trajectory model.
time The name of the time variable.
idents The name of the trajectory identification variable.
nClusters The number of clusters to estimate.
standardize A function to standardize the output matrix of the representation step. By
default, the output is shifted and rescaled to ensure zero mean and unit variance.
... Arguments passed to stats::lm. The following external arguments are ignored:
x, data, control, centers, trace.
See Also


Examples

data(latrendData)
method <- lcMethodLMKM(Y ~ Time, id = "Id", time = "Time", nClusters = 3)
model <- latrend(method, latrendData)

lcMethodLongclust  
Specify Longclust method

Description

Specify Longclust method

Usage

lcMethodLongclust(
  response, 
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  nClusters = 2,
  ...
)

Arguments

response  The name of the response variable.
time  The name of the time variable.
id  The name of the trajectory identifier variable.
nClusters  The number of clusters to estimate.
...  Arguments passed to longclust::longclustEM. The following external arguments are ignored: data, x, Gmin, Gmax, userseed.

See Also

Examples

library(longclust)
data(latrendData)
method <- lcMethodLongclust("Y", id = "Id", time = "Time", nClusters = 3)
model <- latrend(method, latrendData)

Description

Latent profile analysis or finite Gaussian mixture modeling.

Usage

lcMethodMclustLLPA(
  response,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  nClusters = 2,
  ...)

Arguments

response The name of the response variable.
time The name of the time variable.
id The name of the trajectory identifier variable.nClusters The number of clusters to estimate.
... Arguments passed to mclust::Mclust. The following external arguments are ignored: data, G, verbose.

See Also


Examples

library(mclust)
data(latrendData)
method <- lcMethodMclustLLPA("Y", id = "Id", time = "Time", nClusters = 3)
model <- latrend(method, latrendData)
Specify a GLMM with a normal mixture in the random effects

**Usage**

```r
lcMethodMixAK_GLMM(
  fixed,
  random,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  nClusters = 2,
  ...
)
```

**Arguments**

- `fixed`: A formula specifying the fixed effects of the model, including the response. Creates the `y` and `x` arguments for the call to `mixAK::GLMM_MCMC`.
- `random`: A formula specifying the random effects of the model, including the random intercept. Creates the `z` and `random.intercept` arguments for the call to `mixAK::GLMM_MCMC`.
- `time`: The name of the time variable.
- `id`: The name of the trajectory identifier variable. This is used to generate the `id` vector argument for the call to `mixAK::GLMM_MCMC`.
- `nClusters`: The number of clusters.
- `...`: Arguments passed to `mixAK::GLMM_MCMC`. The following external arguments are ignored: `y`, `x`, `z`, `random.intercept`, `silent`.

**See Also**


**Examples**

```r
data(latrendData)
method <- lcMethodMixAK_GLMM(fixed = Y ~ 1, random = ~ Time,
  id = "Id", time = "Time", nClusters = 3)
model <- latrend(method, latrendData)
summary(model)
```
Specify mixed mixture regression model using mixtools

**Usage**

```r
lcMethodMixtoolsGMM(
  formula,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  nClusters = 2,
  ...
)
```

**Arguments**

- `formula`: Formula, including a random effects component for the trajectory. See `lme4::lmer` formula syntax.
- `time`: The name of the time variable.
- `id`: The name of the trajectory identifier variable.
- `nClusters`: The number of clusters.
- `...`: Arguments passed to `mixtools::regmixEM.mixed`. The following arguments are ignored: data, y, x, w, k, addintercept.fixed, verb.

**See Also**


**Examples**

```r
library(mixtools)
library(latrendData)
method <- lcMethodMixtoolsGMM(
  formula = Y ~ Time + (1 | Id),
  id = "Id", time = "Time",
  nClusters = 3,
  arb.R = FALSE)
```
Specify non-parametric estimation for independent repeated measures

Usage

```r
lcMethodMixtoolsNPRM(
  response,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  nClusters = 2,
  blockid = NULL,
  bw = NULL,
  h = NULL,
  ...
)
```

Arguments

- `response`: The name of the response variable.
- `time`: The name of the time variable.
- `id`: The name of the trajectory identifier variable.
- `nClusters`: The number of clusters to estimate.
- `blockid`: See `mixtools::npEM`.
- `bw`: See `mixtools::npEM`.
- `h`: See `mixtools::npEM`.
- `...`: Arguments passed to `mixtools::npEM`. The following optional arguments are ignored: data, x, mu0, verb.

See Also


Examples

```r
library(mixtools)
data(latrendData)
method <- lcMethodMixtoolsNPRM("Y", id = "Id", time = "Time", nClusters = 3)
model <- latrend(method, latrendData)
```
lcMethodMixTVEM

Specify a MixTVEM

Description

Specify a MixTVEM

Usage

lcMethodMixTVEM(
  formula,
  formula.mb = ~1,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  nClusters = 2,
  ...
)

Arguments

formula A formula excluding the time component. Time-invariant covariates are detected automatically as these are a special case in MixTVEM.

formula.mb A formula for cluster-membership prediction. Covariates must be time-invariant. Furthermore, the formula must contain an intercept.

time The name of the time variable.

id The name of the trajectory identifier variable.

nClusters The number of clusters. This replaces the numClasses argument of the TVEMMixNormal function call.

... Arguments passed to the TVEMMixNormal() function. The following optional arguments are ignored: doPlot, getSEs, numClasses.

Note

In order to use this method, you must download and source MixTVEM.R. See the reference below.

References

https://github.com/dziakj1/MixTVEM

Examples

# this example only runs if you download and place MixTVEM.R in your wd
try{
  source("MixTVEM.R")
  method = lcMethodMixTVEM(Value ~ time(1) - 1,
lcMethodRandom

Specify a random-partitioning method

Description

Creates a model with random cluster assignments according to the random cluster proportions drawn from a Dirichlet distribution.

Usage

```r
lcMethodRandom(
  response, 
  alpha = 10, 
  center = meanNA, 
  time = getOption("latrend.time"), 
  id = getOption("latrend.id"), 
  nClusters = 2, 
  name = "random"
)
```

Arguments

- `response` The name of the response variable.
- `alpha` The Dirichlet parameters. Either scalar or of length `nClusters`. The higher alpha, the more uniform the clusters will be.
- `center` Optional function for computing the longitudinal cluster centers, with signature `(x)`.
- `time` The name of the time variable.
- `id` The name of the trajectory identification variable.
- `nClusters` The number of clusters.
- `name` The name of the method.

See Also

Examples

data(latrendData)
method <- lcMethodRandom(response = "Y", id = "Id", time = "Time")
model <- latrend(method, latrendData)

# uniform clusters
method <- lcMethodRandom(alpha = 1e3, nClusters = 3, response = "Y", id = "Id", time = "Time")

# single large cluster
method <- lcMethodRandom(alpha = c(100, 1, 1, 1), nClusters = 4,
                          response = "Y", id = "Id", time = "Time")

---

lcMethods

Generate a list of lcMethod objects

Description

Generates a list of lcMethod objects for all combinations of the provided argument values.

Usage

lcMethods(method, ..., envir = NULL)

Arguments

method  The lcMethod to use as the template, which will be updated for each of the other arguments.

...     Any other arguments to update the lcMethod definition with. Values must be scalar, vector, list, or encapsulated in a .() call. Arguments wrapped in .() are passed as-is to the model call, ensuring a readable method. Arguments comprising a single symbol (e.g. a variable name) are interpreted as a constant. To force evaluation, specify arg=(var) or arg=force(var). Arguments of type vector or list are split across a series of method fit calls. Arguments of type scalar are constant across the method fits. If a list is intended to be passed as a constant argument, then specifying arg=.listObject) results in it being treated as such.

envir   The environment in which to evaluate the method arguments.

Value

A list of lcMethod objects.
Examples

```r
data(latrendData)
baseMethod <- lcMethodKML("Y", id = "Id", time = "Time")
methods <- lcMethods(baseMethod, nClusters = 1:6)

nclus <- 1:6
methods <- lcMethods(baseMethod, nClusters = nclus)

methods <- lcMethods(baseMethod, nClusters = 3, center = .(mean, mean, median))
length(methods) # 3

methods <- lcMethods(baseMethod, nClusters = 1:3, center = .(mean, mean, median))
length(methods) # 9
```

---

**lcMethodStratify**  
*Specify a stratification method*

Description

Specify a stratification method

Usage

```r
lcMethodStratify(
  response,
  stratify,
  center = meanNA,
  nClusters = NaN,
  clusterNames = NULL,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  name = "stratify"
)
```

Arguments

- **response**: The name of the response variable.
- **stratify**: An expression returning a number or factor value per trajectory, representing the cluster assignment. Alternatively, a function can be provided that takes separate trajectory data.frame as input.
- **center**: The function for computing the longitudinal cluster centers, used for representing the cluster trajectories.
- **nClusters**: The number of clusters. This is optional, as this can be derived from the largest assignment number by default, or the number of factor levels.
- **clusterNames**: The names of the clusters. If a factor assignment is returned, the levels are used as the cluster names.
lcMethodTwoStep

Two-step clustering

Description

Feature-based clustering.

time

The name of the time variable.

id

The name of the trajectory identification variable.

name

The name of the method.

See Also


Examples

data(latrendData)
# Stratification based on the mean response level
method <- lcMethodStratify("Y", mean(Y) > 0,
  clusterNames = c("Low", "High"), id = "Id", time = "Time")
model <- latrend(method, latrendData)
summary(model)

# Stratification function
stratfun <- function(trajdata) {
  trajmean <- mean(trajdata$Y)
  factor(trajmean > 1.7,
         levels = c(FALSE, TRUE),
         labels = c("Low", "High"))
}
method <- lcMethodStratify("Y", stratfun, id = "Id", time = "Time")

# Multiple clusters
stratfun3 <- function(trajdata) {
  trajmean <- mean(trajdata$Y)
  cut(trajmean,
      c(-Inf, .5, 2, Inf),
      labels = c("Low", "Medium", "High"))
}
method <- lcMethodStratify("Y", stratfun3, id = "Id", time = "Time")
Usage

```r
lcMethodTwoStep(
  response,
  representationStep,
  clusterStep,
  standardize = scale,
  center = meanNA,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  ...
)
```

Arguments

- `response`: The name of the response variable.
- `representationStep`: A function with signature function(method, data) that computes the representation per strata, returned as a matrix. Alternatively, `representationStep` is a pre-computed representation matrix.
- `clusterStep`: A function with signature function(repdata) that outputs a lcModel.
- `standardize`: A function to standardize the output matrix of the representation step. By default, the output is shifted and rescaled to ensure zero mean and unit variance.
- `center`: Optional function for computing the longitudinal cluster centers, with signature (x).
- `time`: The name of the time variable.
- `id`: The name of the trajectory identification variable.
- `...`: Additional arguments.

See Also


Description

Abstract class for defining estimated longitudinal cluster models.

Extracts the name of the lcModel object. The name is comprised of the underlying lcMethod name, and the assigned label (if any).
Usage

## S4 method for signature 'lcModel'
getLabel(object, ...)

## S4 method for signature 'lcModel'
getName(object)

## S4 method for signature 'lcModel'
getShortName(object)

Arguments

object The lcModel object.
...
Any additional arguments.

Details

An extending class must implement the following methods to ensure basic functionality:

- predict.lcModelExt: Used to obtain the fitted cluster trajectories and trajectories.
- postprob(lcModelExt): The posterior probability matrix is used to determine the cluster assignments of the trajectories.

For predicting the posterior probability for unseen data, the predictPostprob() should be implemented.

Slots

method The lcMethod-class object specifying the arguments under which the model was fitted.
call The call that was used to create this lcModel object. Typically, this is the call to latent() or any of the other fitting functions.
model An arbitrary underlying model representation.
data A data.frame object, or an expression to resolves to the data.frame object.
date The date-time when the model estimation was initiated.
id The name of the trajectory identifier column.
time The name of the time variable.
response The name of the response variable.
label The label assigned to this model.
ids The possible trajectory identifier values the model was fitted on.
clusterNames The names of the clusters.
estimationTime The time, in seconds, that it took to fit the model.
tag An arbitrary user-specified data structure. This slot may be accessed and updated directly.
lcModelCustom

Specify a model based on a pre-computed result.

Description

Specify a model based on a pre-computed result.

Usage

lcModelCustom(
  data,
  response,
  trajectoryAssignments = NULL,
  clusterTrajectories = mean,
  trajectories = data,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  clusterNames = NULL,
  converged = TRUE,
  postprob = NULL,
  model = NULL,
  name = "custom",
  predict = NULL,
  predictPostprob = NULL,
  method = new("lcMethod")
)

Arguments

data The data on which the cluster result is based, a data.frame.
response The response variable.
trajectoryAssignments A vector indicating cluster membership per strata. Either a numeric vector with range 1:numClus, or a factor.
clusterTrajectories The cluster trajectories as a data.frame, or a function computing the center trajectory based on the strata of the respective cluster.
trajectories The fitted trajectories.
time The time variable.
lcModelPartition

id
clusterNames
converged
postprob
model
name
predict
predictPostprob
method

The id variable.
The names of the clusters. Optional.
Convergence state of the model. TRUE by default.
Optional posterior probability matrix.
An optional object representing the internal model.
The name of the model.
Predict function for the response.
Predict function for the posterior probability.
The method used to create this lcModelCustom instance. Optional.

lcModelPartition Create a lcModel with pre-defined partitioning

Description
Represents an arbitrary partitioning of a set of trajectories. As such, this model has no predictive capabilities. The cluster trajectories are represented by the specified center function (mean by default).

Usage

lcModelPartition(
  data,
  response,
  trajectoryAssignments,
  nClusters = NA,
  center = meanNA,
  clusterNames = NULL,
  time =getOption("latrend.time"),
  id =getOption("latrend.id"),
  name = "part",
  envir = parent.frame()
)

Arguments
data A data.frame representing the trajectory data.
response The name of the response variable.
trajectoryAssignments A vector of cluster membership per trajectory, either factor, or integer (1 to nClusters).
nClusters The number of clusters. Optional for factor assignments.
center  The function for computing the longitudinal cluster centers, used for representing the cluster trajectories.

clusterNames  The names of the clusters, or a function with input n outputting a character vector of names.

time  The name of the time variable.

id  The name of the trajectory identification variable.

name  The name of the method.

envir  The environment associated with the model. Used for evaluating the assigned data object by model.data.lcModel.

---

lcModels  *Construct a flat (named) list of lcModel objects*

**Description**

Takes the inputs and generates a named lcModels object containing a list of the input models. Duplicates are preserved.

**Usage**

`lcModels(...)`

**Arguments**

`...`  `lcModel`, `lcModels`, or a recursive list of `lcModel` objects. Arguments may be named.

**Value**

A `lcModels` object containing all specified `lcModel` objects.

**See Also**

Other `lcModel` list functions: `as.lcModels()`, `print.lcModels()`, `subset.lcModels()`

**Examples**

data(latrendData)
kml <- latrend(lcMethodKML("Y", id = "Id", time = "Time"), latrendData)
gmm <- latrend(lcMethodLcmmGMM(Y ~ Time + (1 | Id), id = "Id", time = "Time"), latrendData)
lcModels(kml, gmm)
lcModels(defaults = c(kml, gmm))
lcModelWeightedPartition

Create a lcModel with pre-defined weighted partitioning

Description

Create a lcModel with pre-defined weighted partitioning

Usage

lcModelWeightedPartition(
  data,
  response,
  weights,
  center = weighted.meanNA,
  clusterNames = colnames(weights),
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  name = "wpart"
)

Arguments

data The data on which the cluster result is based, a data.frame.
response The name of the response variable.
weights A numIds x numClusters matrix of partition probabilities.
center The function for computing the longitudinal cluster centers, used for representing the cluster trajectories.
clusterNames The names of the clusters, or a function with input n outputting a character vector of names.
time The name of the time variable.
id The name of the trajectory identification variable.
name The name of the method.

logLik.lcModel

Extract the log-likelihood of a lcModel

Description

Extract the log-likelihood of a lcModel
max.lcModels

Usage

## S3 method for class 'lcModel'
logLik(object, ...)

Arguments

object The lcModel object.

... Additional arguments.

See Also

Other model-specific methods: `clusterTrajectories()`, `coef.lcModel()`, `converged()`, `deviance.lcModel()`,
df.residual.lcModel(), fitted.lcModel(), lcModel-class, model.frame.lcModel(), nobs.lcModel(),
postprob(), predict.lcModel(), predictAssignments(), predictForCluster(), predictPostprob(),
residuals.lcModel(), sigma.lcModel(), time.lcModel(), trajectories()

max.lcModels Select the lcModel with the highest metric value

Description

Select the lcModel with the highest metric value

Usage

## S3 method for class 'lcModels'
max(x, name, ...)

Arguments

x The lcModels object.

name The name of the internal metric.

... Additional arguments.

Value

The lcModel with the highest metric value

See Also

`min.lcModels` externalMetric
Examples

```r
data(latrendData)
baseMethod <- lcMethodKML(response = "Y", id = "Id", time = "Time")
kml1 <- latrend(baseMethod, nClusters = 1, latrendData)
kml2 <- latrend(baseMethod, nClusters = 2, latrendData)
kml3 <- latrend(baseMethod, nClusters = 3, latrendData)
models <- lcModels(kml1, kml2, kml3)
max(models, '/WRSS/')
```

---

meltRepeatedMeasures  Convert a repeated measures data matrix to a data.frame

Description

Convert a repeated measures data matrix to a data.frame

Usage

```r
meltRepeatedMeasures(
  data,
  response,
  id = getOption("latrend.id"),
  time = getOption("latrend.time"),
  ids = rownames(data),
  times = colnames(data),
  as.data.table = FALSE
)
```

Arguments

- `data` The matrix containing a trajectory on each row.
- `response` The response column name.
- `id` The id column name.
- `time` The time column name.
- `ids` A vector specifying the id names. Should match the number of rows of `data`.
- `times` A numeric vector specifying the times of the measurements. Should match the number of columns of `data`.
- `as.data.table` Whether to return the result as a `data.table`, or a `data.frame` otherwise.

Value

A `data.table` or `data.frame` containing the repeated measures.
metric

Compute internal model metric(s)

Description
Compute internal model metric(s)

Usage

## S4 method for signature 'lcModel'
metric(object, name = c("AIC", "BIC", "WRSS", "APPA"), ...)

## S4 method for signature 'list'
metric(object, name, drop = TRUE)

## S4 method for signature 'lcModels'
metric(object, name, drop = TRUE)

Arguments

object The lcModel, lcModels, or list of lcModel objects to compute the metrics for.
name The name(s) of the metric(s) to compute.
... Additional arguments.
drop Whether to drop the matrix dimensions in case of a single model output.

Value
For metric(lcModel): A named numeric vector with the computed model metrics.
For metric(list): A data.frame with a metric per column.
For metric(lcModels): A data.frame with a metric per column.

See Also

externalMetric min.lcModels max.lcModels
Other metric functions: defineExternalMetric(), defineInternalMetric(), externalMetric, lcModel, lcModel-method, getExternalMetricDefinition(), getExternalMetricNames(), getInternalMetricDefinition(), getInternalMetricNames()

Examples

data(latrendData)
model <- latrend(lcMethodLcmmGMM(Y ~ Time + (1 | Id), id = "Id", time = "Time"), latrendData)
bic <- metric(model, "BIC")

ic <- metric(model, c("AIC", "BIC"))
Select the lcModel with the lowest metric value

## S3 method for class 'lcModels'
min(x, name, ...)

Arguments
- `x`: The lcModels object
- `name`: The name of the internal metric.
- `...`: Additional arguments.

Value
The lcModel with the lowest metric value

See Also
- `max.lcModels`
- `externalMetric`

Examples
```r
data(latrendData)
baseMethod <- lcMethodKML(response = "Y", id = "Id", time = "Time")
kml1 <- latrend(baseMethod, nClusters = 1, latrendData)
kml2 <- latrend(baseMethod, nClusters = 2, latrendData)
kml3 <- latrend(baseMethod, nClusters = 3, latrendData)
models <- lcModels(kml1, kml2, kml3)
min(models, 'WRSS')
```

Extract the model data that was used for fitting

## S3 method for class 'lcModel'
model.data(object, ...)

Description
Evaluates the data call in the environment that the model was trained in.

Usage
```r
## S3 method for class 'lcModel'
model.data(object, ...)
```
Extract model training data

### Description

Extract model training data

### Usage

```r
## S3 method for class 'lcModel'
model.frame(formula, ...)  
```

### Arguments

- **formula**
  - The lcModel object.
- **...**
  - Additional arguments.

### See Also

Other model-specific methods: `clusterTrajectories()`, `coef.lcModel()`, `converged()`, `deviance.lcModel()`, `df.residual.lcModel()`, `fitted.lcModel()`, `lcModel-class`, `logLik.lcModel()`, `nobs.lcModel()`, `postprob()`, `predict.lcModel()`, `predictAssignments()`, `predictForCluster()`, `predictPostprob()`, `residuals.lcModel()`, `sigma.lcModel()`, `time.lcModel()`, `trajectories()`

---

### nClusters

**Number of clusters**

### Description

Number of clusters

### Usage

```r
nClusters(object)
```

### Arguments

- **object**
  - The lcModel object.
### nIds

<table>
<thead>
<tr>
<th>nIds</th>
<th>Number of strata</th>
</tr>
</thead>
</table>

**Description**

Number of strata

**Usage**

nIds(object)

**Arguments**

- **object**: The lcModel object.

### nobs.lcModel

Extract the number of observations from a lcModel

**Description**

Extract the number of observations from a lcModel

**Usage**

```
## S3 method for class 'lcModel'
nobs(object, ...)
```

**Arguments**

- **object**: The lcModel object.
- **...**: Additional arguments.

**See Also**

Other model-specific methods: `clusterTrajectories()`, `coef.lcModel()`, `converged()`, `deviance.lcModel()`, `df.residual.lcModel()`, `fitted.lcModel()`, `lcModel-class`, `logLik.lcModel()`, `model.frame.lcModel()`, `postprob()`, `predict.lcModel()`, `predictAssignments()`, `predictForCluster()`, `predictPostprob()`, `residuals.lcModel()`, `sigma.lcModel()`, `time.lcModel()`, `trajectories()`
**plotClusterTrajectories**

**Plot a lcModel**

**Description**

Plot a lcModel. By default, this plots the cluster trajectories of the model, along with the training data.

**Usage**

```r
## S4 method for signature 'lcModel,ANY'
plot(x, y, ...)
```

**Arguments**

- `x`: The lcModel object.
- `y`: Not used.
- `...`: Arguments passed to `plotClusterTrajectories`.

**Value**

A ggplot object.

---

**plotClusterTrajectories**

**Plot cluster trajectories**

**Description**

Plot cluster trajectories

Plot the cluster trajectories of a lcModel

**Usage**

```r
## S4 method for signature 'data.frame'
plotClusterTrajectories(
  object,
  response,
  cluster = "Cluster",
  time = getOption("latrend.time"),
  center = meanNA,
  trajectories = FALSE,
  facet = trajectories,
  id = getOption("latrend.id"),
)```
## S4 method for signature 'lcModel'
plotClusterTrajectories(
  object,
  what = "mu",
  at = time(object),
  clusterLabels = sprintf("%s (%s)", clusterNames(object),
                        percent(clusterProportions(object))),
  trajectories = FALSE,
  facet = trajectories,
  trajAssignments = trajectoryAssignments(object),
  ...
)

### Arguments

- **object**: The (cluster) trajectory data.
- **response**: The name of the response variable.
- **cluster**: The cluster assignment column.
- **time**: The name of the time variable.
- **center**: A function for aggregating multiple points at the same point in time.
- **trajectories**: Whether to plot the original data in addition to the cluster (i.e., center) trajectories.
- **facet**: Whether to facet by cluster. This is done by default when trajectories is enabled.
- **id**: Id column. Only needed when trajectories = TRUE.
- **...**: Arguments passed to `clusterTrajectories`.
- **what**: The distributional parameter to predict. By default, the mean response ‘mu’ is predicted. The cluster membership predictions can be obtained by specifying what='mb'.
- **at**: An optional vector, list or data frame of covariates at which to compute the cluster trajectory predictions. If a vector is specified, this is assumed to be the time covariate. Otherwise, a named list or data frame must be provided.
- **clusterLabels**: Cluster display names. By default it’s the cluster name with its proportion enclosed in parentheses.
- **trajAssignments**: The cluster assignments for the fitted trajectories. Only used when trajectories = TRUE and facet = TRUE. See `trajectoryAssignments`.

### Details

The line size of the cluster trajectories can be manipulated by changing the default for ggplot2. For example: `update_geom_defaults("line",list(size = 1.5))`
plotMetric

Plot one or more internal metrics for all lcModels

Description

Plot one or more internal metrics for all lcModels

Usage

plotMetric(models, name, by = "nClusters", subset, group = character())

Arguments

models A lcModels or list of lcModel objects to compute and plot the metrics of.
name The name(s) of the metric(s) to compute.
by The argument name along which methods are plotted.
subset Logical expression based on the lcModel method arguments, indicating which lcModel objects to keep.
group The argument names to use for determining groups of different models. By default, all arguments are included. Specifying group = character() disables grouping. Specifying a single argument for grouping uses that specific column as the grouping column. In all other cases, groupings are represented by a number.

Value

ggplot2 object.

Examples

data(latrendData)
baseMethod <- lcMethodKML(response = "Y", id = "Id", time = "Time")
kml1 <- latrend(baseMethod, nClusters = 1, latrendData)
kml2 <- latrend(baseMethod, nClusters = 2, latrendData)
kml3 <- latrend(baseMethod, nClusters = 3, latrendData)
models <- lcModels(kml1, kml2, kml3)
plotMetric(models, "BIC", by = "nClusters", group = ".name")
plotTrajectories

Plot trajectories
Plot fitted trajectories of a lcModel

Usage

## S4 method for signature 'data.frame'
plotTrajectories(
  object,
  response,
  time = getOption("latrend.time"),
  id = getOption("latrend.id"),
  cluster = NULL,
  facet = TRUE,
  ...
)

## S4 method for signature 'lcModel'
plotTrajectories(object, ...)

Arguments

- **object**  The model.
- **response** Response variable character name or a call.
- **time** The name of the time variable.
- **id** The name of the trajectory identifier variable.
- **cluster** Cluster variable name. If unspecified, trajectories are not grouped. Alternatively, cluster is a vector indicating cluster membership per id.
- **facet** Whether to facet by cluster.
- **...** Arguments passed on to `trajectories`

- **at** The time points at which to compute the id-specific trajectories.
- **what** The distributional parameter to compute the response for.
- **clusters** The cluster assignments for the strata to base the trajectories on.

Examples

data(latrendData)
plotTrajectories(latrendData, response = "Y", id = "Id", time = "Time")

plotTrajectories(latrendData, response = quote(exp(Y)), id = "Id", time = "Time")
postprobFromAssignments

postprob  

Posterior probability per fitted id

Description

Posterior probability per fitted id

Usage

## S4 method for signature 'lcModel'
postprob(object, ...)

Arguments

object  
The lcModel.

...  
Additional arguments.

See Also

Other model-specific methods: clusterTrajectories(), coef.lcModel(), converged(), deviance.lcModel(), df.residual.lcModel(), fitted.lcModel(), lcModel-class, logLik.lcModel(), model.frame.lcModel(), nobs.lcModel(), predict.lcModel(), predictAssignments(), predictForCluster(), predictPostprob(), residuals.lcModel(), sigma.lcModel(), time.lcModel(), trajectories()

Examples

data(latrendData)
model <- latrend(lcMethodLcmmGMM(Y ~ Time + (1 | Id), id = "Id", time = "Time"), data = latrendData)
postprob(model)

postprobFromAssignments

Create a posterior probability matrix from a vector of cluster assignments.

Description

For each trajectory, the probability of the assigned cluster is 1.

Usage

postprobFromAssignments(assignments, k)

Arguments

assignments  
Integer vector indicating cluster assignment per trajectory

k  
The number of clusters.
predict.lcModel  lcModel predictions

Description

Predicts the expected trajectory observations at the given time for each cluster, unless specified.

Usage

```r
## S3 method for class 'lcModel'
predict(object, ..., newdata = NULL, what = "mu")
```

Arguments

- `object`: The lcModel object.
- `...`: Additional arguments.
- `newdata`: Optional data frame for which to compute the model predictions. If omitted, the model training data is used. Cluster trajectory predictions are made when ids are not specified. If the clusters are specified under the Cluster column, output is given only for the specified cluster. Otherwise, a matrix is returned with predictions for all clusters.
- `what`: The distributional parameter to predict. By default, the mean response 'mu' is predicted. The cluster membership predictions can be obtained by specifying what='mb'.

Details

The default `predict.lcModel` implementation.

Value

If `newdata` specifies the cluster membership; a vector of cluster-specific predictions. Otherwise, a matrix of predictions is returned corresponding to each cluster.

See Also

Other model-specific methods: `clusterTrajectories()`, `coef.lcModel()`, `converged()`, `deviance.lcModel()`, `df.residual.lcModel()`, `Fitted.lcModel()`, `lcModel-class`, `logLik.lcModel()`, `model.frame.lcModel()`, `mobs.lcModel()`, `postprob()`, `predictAssignments()`, `predictForCluster()`, `predictPostprob()`, `residuals.lcModel()`, `sigma.lcModel()`, `time.lcModel()`, `trajectories()`
predictAssignments

Predict the cluster assignments for new trajectories

Description

Computes the posterior probability based on the provided (observed) data.

Usage

## S4 method for signature 'lcModel'
predictAssignments(object, newdata = NULL, strategy = which.max, ...)

Arguments

- **object**: The lcModel object.
- **newdata**: Optional data frame for which to compute the model predictions. If omitted, the model training data is used. Cluster trajectory predictions are made when ids are not specified. If the clusters are specified under the Cluster column, output is given only for the specified cluster. Otherwise, a matrix is returned with predictions for all clusters.
- **strategy**: A function returning the cluster index based on the given vector of membership probabilities. By default, ids are assigned to the cluster with the highest probability.
- **...**: Additional arguments.

Details

The default implementation uses `predictPostprob` to determine the cluster membership.

Value

A factor with length `nrow(newdata)` that indicates the posterior probability per trajectory per observation.
predictForCluster

Description

Predicts the expected trajectory observations at the given time under the assumption that the trajectory belongs to the specified cluster.

Usage

## S4 method for signature 'lcModel'
predictForCluster(object, newdata = NULL, cluster, ..., what = "mu")

Arguments

- **object**: The lcModel object.
- **newdata**: Optional data frame for which to compute the model predictions. If omitted, the model training data is used. Cluster trajectory predictions are made when ids are not specified. If the clusters are specified under the Cluster column, output is given only for the specified cluster. Otherwise, a matrix is returned with predictions for all clusters.
- **cluster**: The cluster name (as character) to predict for.
- **...**: Additional arguments.
- **what**: The distributional parameter to predict. By default, the mean response 'mu' is predicted. The cluster membership predictions can be obtained by specifying what='mb'.

Value

A vector with the predictions per newdata observation, or a data.frame with the predictions and newdata alongside.

See Also

predict.lcModel

Other model-specific methods: clusterTrajectories(), coef.lcModel(), converged(), deviance.lcModel(), df.residual.lcModel(), fitted.lcModel(), lcModel-class, logLik.lcModel(), model.frame.lcModel(), nobs.lcModel(), postprob(), predict.lcModel(), predictForCluster(), predictPostprob(), residuals.lcModel(), sigma.lcModel(), time.lcModel(), trajectories()
predictPostprob  

lcModel posterior probability prediction

Description

lcModel posterior probability prediction

Usage

```r
## S4 method for signature 'lcModel'
predictPostprob(object, newdata = NULL, ...)
```

Arguments

- `object`: The lcModel to predict the posterior probabilities with.
- `newdata`: Optional data frame for which to compute the posterior probability. If omitted, the model training data is used.
- `...`: Additional arguments.

Details

The default implementation returns a uniform probability matrix.

Value

A matrix indicating the posterior probability per trajectory per measurement on each row, for each cluster (the columns).

See Also

Other model-specific methods: `clusterTrajectories()`, `coef.lcModel()`, `converged()`, `deviance.lcModel()`, `df.residual.lcModel()`, `fitted.lcModel()`, `lcModel-class`, `logLik.lcModel()`, `model.frame.lcModel()`, `mobs.lcModel()`, `postprob()`, `predict.lcModel()`, `predictAssignments()`, `predictForCluster()`, `residuals.lcModel()`, `sigma.lcModel()`, `time.lcModel()`, `trajectories()`

print.lcMethod  

Print the arguments of an lcMethod object

Description

Print the arguments of an lcMethod object

Usage

```r
## S3 method for class 'lcMethod'
print(x, ..., eval = FALSE, width = 40, envir = NULL)
```
print.lcModels

Arguments

x The lcMethod object.
...
not used.
eval Whether to print the evaluated argument values.
width Maximum number of characters per argument.
eenv The environment in which to evaluate the arguments when eval = TRUE.

Description

Print lcModels list concisely

Usage

## S3 method for class 'lcModels'
print(
  x,
  ..., 
  summary = FALSE,
  excludeShared = !getOption("latrend.printSharedModelArgs")
)

Arguments

x The lcModels object.
...
not used.
snmary Whether to print the complete summary per model. This may be slow for long lists!
excludeShared Whether to exclude model arguments which are identical across all models.

See Also

Other lcModel list functions: as.lcModels(), lcModels, subset.lcModels()
qqPlot  

Quantile-quantile plot

Description
Quantile-quantile plot

Usage
## S4 method for signature 'lcModel'
qqPlot(object, byCluster = FALSE, ...)

Arguments

- **object**: The model.
- **byCluster**: Whether to plot the Q-Q line per cluster
- **...**: Other arguments passed to ggplot2::geom_qq_band, ggplot2::stat_qq_line, and ggplot2::stat_qq_point.

See Also
residuals.lcModel metric plotClusterTrajectories

residuals.lcModel  

Extract lcModel residuals

Description
Extract lcModel residuals

Usage
## S3 method for class 'lcModel'
residuals(object, ..., clusters = trajectoryAssignments(object))

Arguments

- **object**: The lcModel object.
- **...**: Additional arguments.
- **clusters**: Optional cluster assignments per id. If unspecified, a matrix is returned containing the cluster-specific predictions per column.

Value
A vector of residuals for the cluster assignments specified by clusters. If clusters is unspecified, a matrix of cluster-specific residuals per observations is returned.
responseVariable

See Also

Other model-specific methods: clusterTrajectories(), coef.lcModel(), converged(), deviance.lcModel(), df.residual.lcModel(), fitted.lcModel(), lcModel-class, logLik.lcModel(), model.frame.lcModel(), nobs.lcModel(), postprob(), predict.lcModel(), predictAssignments(), predictForCluster(), predictPostprob(), sigma.lcModel(), time.lcModel(), trajectories()

responseVariable  Extract the response variable

Description

Extracts the response variable from the given object.

Usage

## S4 method for signature 'lcMethod'
responseVariable(object, ...)

## S4 method for signature 'lcModel'
responseVariable(object, ...)

Arguments

object  The object to extract the response variable from.
...
Additional arguments.

Details

If the lcMethod object specifies a formula argument, then the response is extracted from the response term of the formula.

Value

The response variable name as a character.

See Also

Other lcModel variables: idVariable(), timeVariable()

Examples

method <- lcMethodKML("Value")
responseVariable(method) # "Value"

method <- lcMethodLcmmGBTM(Value ~ Time)
responseVariable(method) # "Value"

data(latrendData)
model <- latrend(lcMethodKML("Y", id = "Id", time = "Time"), latrendData)
responseVariable(model) # "Value"

sigma.lcModel

Extract residual standard deviation from a lcModel

Description
Extract residual standard deviation from a lcModel

Usage
### S3 method for class 'lcModel'
sigma(object, ...)

Arguments

  object      The lcModel object.
  ...         Additional arguments.

See Also
Other model-specific methods: clusterTrajectories(), coef.lcModel(), converged(), deviance.lcModel(),
df.residual.lcModel(), fitted.lcModel(), lcModel-class, logLik.lcModel(), model.frame.lcModel(),
nobs.lcModel(), postprob(), predict.lcModel(), predictAssignments(), predictForCluster(),
predictPostprob(), residuals.lcModel(), time.lcModel(), trajectories()

strip

Strip a lcModel for serialization

Description
Removes associated environments from any of the arguments. This is typically the case for arguments of type formula.
Strip a lcModel of non-essential variables and environments in order to reduce the model size for serialization.

Usage
### S4 method for signature 'lcMethod'
strip(object, ...)

### S4 method for signature 'lcModel'
strip(object, ...)
subset.lcModels

Arguments

object
The lcModel.

... Additional arguments.

Description

Subsetting a lcModels list based on method arguments

Usage

## S3 method for class 'lcModels'
subset(x, subset, drop = FALSE, ...)

Arguments

x
The lcModels or list of lcModel to be subsetted.

subset Logical expression based on the lcModel method arguments, indicating which lcModel objects to keep.

drop Whether to return a lcModel object if the result is length 1.

... Not used.

Value

A lcModels list with the subset of lcModel objects.

See Also

Other lcModel list functions: as.lcModels(), lcModels, print.lcModels()

Examples

data(latrendData)
mKML <- lcMethodKML(response = "Y", id = "Id", time = "Time")
kml1 <- latrend(mKML, nClusters = 1, latrendData)
kml2 <- latrend(mKML, nClusters = 2, latrendData)
kml3 <- latrend(mKML, nClusters = 3, latrendData)
gmm <- latrend(lcMethodLcmmGMM(Y ~ Time + (1 | Id), id = "Id", time = "Time"), latrendData)
models <- lcModels(kml1, kml2, kml3, gmm)
subset(models, nClusters > 1 & .method == 'kml')
### summary.lcModel

**Summarize a lcModel**

**Description**

Extracts all relevant information from the underlying model into a list.

**Usage**

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

**Arguments**

- `object`  
The `lcModel` object.
- `...`  
  Additional arguments.

### time.lcModel

**Sampling times of a lcModel**

**Description**

Sampling times of a lcModel.

**Usage**

```r
## S3 method for class 'lcModel'
time(x, ...)
```

**Arguments**

- `x`  
The `lcModel` object.
- `...`  
  Not used.

**Value**

The unique times at which observations occur.

**See Also**

Other model-specific methods: `clusterTrajectories()`, `coef.lcModel()`, `converged()`, `deviance.lcModel()`, `df.residual.lcModel()`, `fitted.lcModel()`, `lcModel-class`, `logLik.lcModel()`, `model.frame.lcModel()`, `nobs.lcModel()`, `postprob()`, `predict.lcModel()`, `predictAssignments()`, `predictForCluster()`, `predictPostprob()`, `residuals.lcModel()`, `sigma.lcModel()`, `trajectories()`
timeVariable

Extract the time variable

Description

Extracts the time variable (i.e., column name) from the given object.

Usage

```r
## S4 method for signature 'lcMethod'
getTimeVariable(object, ...)

## S4 method for signature 'lcModel'
getTimeVariable(object)
```

Arguments

- `object`: The object to extract the variable from.
- `...`: Additional arguments.

Value

The time variable name, as character.

See Also

Other lcModel variables: `idVariable()`, `responseVariable()`

Examples

```r
method <- lcMethodKML(time = "Assessment")
timeVariable(method) # "Assessment"

data(latrendData)
model <- latrend(lcMethodKML("Y", id = "Id", time = "Time"), latrendData)
idVariable(model) # "Id"
```

trajectories

Extract the fitted trajectories for all strata

Description

Extract the fitted trajectories for all strata
trajectories

Usage

trajectories(
  object,
  at = time(object),
  what = "mu",
  clusters = trajectoryAssignments(object),
  ...
)

## S4 method for signature 'lcModel'

trajectories(
  object,
  at = time(object),
  what = "mu",
  clusters = trajectoryAssignments(object),
  ...
)

Arguments

- **object**: The model.
- **at**: The time points at which to compute the id-specific trajectories.
- **what**: The distributional parameter to compute the response for.
- **clusters**: The cluster assignments for the strata to base the trajectories on.
- **...**: Additional arguments.

See Also

- Other model-specific methods: `clusterTrajectories()`, `coef.lcModel()`, `converged()`, `deviance.lcModel()`, `df.residual.lcModel()`, `fitted.lcModel()`, `lcModel-class`, `logLik.lcModel()`, `model.frame.lcModel()`, `nobs.lcModel()`, `postprob()`, `predict.lcModel()`, `predictAssignments()`, `predictForCluster()`, `predictPostprob()`, `residuals.lcModel()`, `sigma.lcModel()`, `time.lcModel()`

Examples

```r
data(latrendData)
model <- latrend(method = lcMethodKML("Y", id = "Id", time = "Time"), data = latrendData)
trajectories(model)

trajectories(model, at = c(0, .5, 1))
```
trajectoryAssignments  Get the cluster membership of each trajectory

Description

Get the cluster membership of each trajectory

Usage

```r
## S4 method for signature 'lcModel'
trajectoryAssignments(object, strategy = which.max, ...)
```

Arguments

- **object**: The lcModel to obtain the cluster assignments from.
- **strategy**: A function returning the cluster index based on the given vector of membership probabilities. By default, ids are assigned to the cluster with the highest probability.
- **...**: Any additional arguments passed to the strategy function.

Details

While the default strategy is `which.max`, it is recommended to use `which.is.max` instead, as this function breaks ties randomly. Another strategy to consider is the function `which.weight`, which enables weighted sampling of cluster assignments.

Examples

```r
data(latrendData)
model <- latrend(method = lcMethodKML("Y", id = "Id", time = "Time"), latrendData)
trajectoryAssignments(model)

# assign ids at random using weighted sampling
trajectoryAssignments(model, strategy = which.weight)
```

transformFitted  Helper function for ensuring the right fitted() output

Description

This function is also responsible for checking whether the input data is valid, such that the fitting process can fail early.
transformLatrendData

Usage

transformFitted(pred, model, clusters)

## S4 method for signature 'NULL,lcModel'
transformFitted(pred, model, clusters)

## S4 method for signature 'matrix,lcModel'
transformFitted(pred, model, clusters)

## S4 method for signature 'list,lcModel'
transformFitted(pred, model, clusters)

## S4 method for signature 'data.frame,lcModel'
transformFitted(pred, model, clusters)

Arguments

pred Prediction object
model The model from which the prediction is made.
clusters Optional argument for specifying the trajectory cluster assignments.

Value

A vector if the clusters argument is specified, else a matrix with the fitted values per cluster per column.

transformLatrendData Transform latrend input data into the right format

Description

This function is also responsible for checking whether the input data is valid, such that the fitting process can fail early.

Usage

transformLatrendData(object, id, time, response, envir)

## S4 method for signature 'data.frame'
transformLatrendData(object, id, time, response, envir)

## S4 method for signature 'matrix'
transformLatrendData(object, id, time, response, envir)

## S4 method for signature 'call'
transformLatrendData(object, id, time, response, envir)
transformPredict

Arguments

object
id
time
response
envir

The data object to transform.
The name of the trajectory identifier variable.
The name of the time variable.
The name of the response variable.
The environment used to evaluate the data object in (e.g., in case object is of type call).

Value

A data.frame with an id, time, and measurement columns.

transformPredict

Helper function that matches the output to the specified newdata

Description

If Cluster is not provided, the prediction is outputted in long format per cluster, resulting in a longer data.frame than the newdata input

Usage

transformPredict(pred, model, newdata)

## S4 method for signature 'NULL',lcModel
transformPredict(pred, model, newdata)

## S4 method for signature 'vector',lcModel
transformPredict(pred, model, newdata)

## S4 method for signature 'matrix',lcModel
transformPredict(pred, model, newdata)

## S4 method for signature 'data.frame',lcModel
transformPredict(pred, model, newdata)

Arguments

pred
model
newdata

The prediction object
The model for which the prediction is made.
A data.frame containing the input data to predict for.

Value

A data.frame with the predictions, or a list of cluster-specific prediction frames
Description

Update a method specification

Usage

```r
## S3 method for class 'lcMethod'
update(object, ..., .eval = FALSE, .remove = character(), envir = NULL)
```

Arguments

- `object` The lcMethod object.
- `...` The new or updated method argument values.
- `.eval` Whether to assign the evaluated argument values to the method. By default (FALSE), the argument expression is preserved.
- `.remove` Names of arguments that should be removed.
- `envir` The environment in which to evaluate the arguments. If NULL, the environment associated with the object is used. If not available, the parent.frame() is used.

Details

Updates or adds arguments to a lcMethod object. The inputs are evaluated in order to determine the presence of formula objects, which are updated accordingly.

Value

The new lcMethod object with the additional or updated arguments.

See Also

Other lcMethod functions: `[[.lcMethod-method,as.data.frame.lcMethods(),as.data.frame.lcMethod(),as.lcMethods(),as.list.lcMethod(),evaluate.lcMethod(),formula.lcMethod(),lcMethod-class`

Examples

```r
m <- lcMethodLcmmbGBTM(Value ~ 1)
m2 <- update(m, formula = ~ . + Time)
m3 <- update(m2, nClusters = 3)
k <- 2
m4 <- update(m, nClusters = k) # nClusters: k
m5 <- update(m, nClusters = k, .eval = TRUE) # nClusters: 2
```
update.lcModel  

Update a lcModel

Description

Fit a new model with modified arguments from the current model.

Usage

## S3 method for class 'lcModel'
update(object, ...)

Arguments

- **object**  
The lcModel object.
- **...**  
Arguments passed on to latrend method.
- **method**  
The lcMethod object specifying the longitudinal cluster method to apply.
- **data**  
The data.frame or matrix to which to apply the method.
- **envir**  
The environment in which to evaluate the method arguments. Note that
  this only applies to data when data is a call.
- **verbose**  
The level of verbosity. Either an object of class Verbose (see R.utils::Verbose  
  for details), a logical indicating whether to show basic computation information, a numeric  
  indicating the verbosity level (see Verbose), or one of c('info','fine','finest').

which.weight  

Sample an index of a vector weighted by the elements

Description

Returns a random index, weighted by the element magnitudes. This function is intended to be used
as an optional strategy for trajectoryAssignments, resulting in randomly sampled cluster membership.

Usage

which.weight(x)

Arguments

- **x**  
A positive numeric vector.
Value

An integer giving the index of the sampled element.

Examples

x = c(.01, .69, .3)
which.weight(x) #1, 2, or 3

[[,lcMethod-method Retrieve and evaluate a lcMethod argument by name

Description

Retrieve and evaluate a lcMethod argument by name

Usage

## S4 method for signature 'lcMethod'
x[[i, eval = TRUE, envir = NULL]]

Arguments

x The lcMethod object.
i Name or index of the argument to retrieve.
eval Whether to evaluate the call argument (enabled by default).
evir The environment in which to evaluate the argument. This argument is only applicable when eval = TRUE.

Value

The argument call or evaluation result.

See Also

Other lcMethod functions: as.data.frame.lcMethods(), as.data.frame.lcMethod(), as.lcMethods(), as.list.lcMethod(), evaluate.lcMethod(), formula.lcMethod(), lcMethod-class, update.lcMethod()

Examples

m = lcMethodKML(nClusters = 5)
m[["nClusters"]"
# 5

k = 2
m = lcMethodKML(nClusters = k)
m[["nClusters", eval=FALSE]] # k
Description

Retrieve and evaluate a lcMethod argument by name

Usage

```r
## S4 method for signature 'lcMethod'
x$name
```

Arguments

- `x` The lcMethod object.
- `name` Name of the argument to retrieve.

Value

The argument evaluation result.

Examples

```r
m <- lcMethodKML(nClusters = 3)
m$nClusters # 3
```
Index

* datasets
  latrendData, 33

* lcMethod functions
  [[,lcMethod-method, 96
  as.data.frame.lcMethod, 6
  as.data.frame.lcMethods, 6
  as.lcMethods, 8
  as.list.lcMethod, 9
  evaluate.lcMethod, 20
  formula.lcMethod, 23
  lcMethod-class, 36
  update.lcMethod, 94

* lcMethod implementations
  lcMethod-class, 36
  lcMethodAKMedoids, 40
  lcMethodCrimCV, 41
  lcMethodCustom, 42
  lcMethodDtwclust, 43
  lcMethodFunFEM, 46
  lcMethodGCKM, 47
  lcMethodKML, 48
  lcMethodLcmmGBTM, 49
  lcMethodLcmmGMM, 50
  lcMethodLKM, 51
  lcMethodLongclust, 52
  lcMethodMclustLLPA, 53
  lcMethodMixAK_GLM, 54
  lcMethodMixtoolsGMM, 55
  lcMethodMixtoolsNPRM, 56
  lcMethodRandom, 58
  lcMethodStratify, 60
  lcMethodTwoStep, 61

* lcMethod package interfaces
  lcMethodFlexmix, 44
  lcMethodFlexmixGBTM, 45

* lcModel list functions
  as.lcModels, 8
  lcModels, 66
  print.lcModels, 83
  subset.lcModels, 87

* lcModel variables
  idVariable, 28
  responseVariable, 85
  timeVariable, 89

* longitudinal cluster fit functions
  latrend, 29
  latrendBatch, 30
  latrendBoot, 31
  latrendCV, 32
  latrendRep, 34

* metric functions
  defineExternalMetric, 17
  defineInternalMetric, 18
  externalMetric,lcModel,lcModel-method, 21
  getExternalMetricDefinition, 25
  getExternalMetricNames, 26
  getInternalMetricDefinition, 26
  getInternalMetricNames, 27
  metric, 70

* model-specific methods
  clusterTrajectories, 12
  coef.lcModel, 13
  converged, 14
  deviance.lcModel, 19
  df.residual.lcModel, 19
  fitted.lcModel, 22
  lcModel-class, 62
  logLik.lcModel, 67
  model.frame.lcModel, 72
  nobs.lcModel, 73
  postprob, 78
  predict.lcModel, 79
  predictAssignments, 80
  predictForCluster, 81
  predictPostprob, 82
  residuals.lcModel, 84
  sigma.lcModel, 86
time.lcModel, 88
trajectories, 89

* **validation methods**
  createTestDataFold, 15
  createTestDataFolds, 15
  createTrainDataFolds, 16
  latrendBoot, 31
  latrendCV, 32

[[, lcMethod-method, 96
$,$lcMethod-method, 97

_PACKAGE (latrend-package), 5

akmedoids::akmedoids.clstr, 40
as.data.frame.lcMethod, 6, 7–9, 21, 23, 38, 94, 96
as.data.frame.lcMethods, 6, 6, 8, 9, 21, 23, 38, 94, 96
as.data.frame.lcModels, 7
as.lcMethods, 6, 7, 8, 9, 21, 23, 38, 94, 96
as.lcModels, 8, 66, 83, 87
as.list.lcMethod, 6–8, 9, 21, 23, 38, 94, 96
atomic, 6

clusterNames, 10
clusterNames<-, 10
clusterProportions, 11
clusterProportions, lcMethod-method
(clusterProportions), 11

clusterSizes, 11
clusterTrajectories, 12, 13, 14, 19, 22, 64, 68, 72, 73, 75, 78, 79, 81, 82, 85, 86, 88, 90
clusterTrajectories, lcModel-method
(clusterTrajectories), 12
coeff.lcModel, 12, 13, 14, 19, 22, 64, 68, 72, 73, 78, 79, 81, 82, 85, 86, 88, 90

compose (lcMethod-class), 36
compose, lcMethod-method
(lcMethod-class), 36

confusionMatrix, 13
converged, 12, 13, 14, 19, 22, 64, 68, 72, 73, 78, 79, 81, 82, 85, 86, 88, 90

converged, lcModel-method (converged), 14
createTestDataFold, 15, 15, 16, 32, 33
createTestDataFolds, 15, 15, 16, 32, 33
createTrainDataFolds, 15, 16, 16, 32, 33
crimCV::crimCV, 41
dcastRepeatedMeasures, 17

defineExternalMetric, 17, 18, 22, 26, 27, 70
defineInternalMetric, 18, 18, 22, 26, 27, 70
deviance.lcModel, 12–14, 19, 19, 22, 64, 68, 72, 73, 78, 79, 81, 82, 85, 86, 88, 90
df.residual.lcModel, 12–14, 19, 19, 22, 64, 68, 72, 73, 78, 79, 81, 82, 85, 86, 88, 90
dist, 21
dtwclust::tsclust, 43
evaluationTime, 20
evaluate.lcMethod, 6–9, 20, 23, 38, 94, 96
externalMetric, 68, 70, 71

externalMetric
(externalMetric, lcModel, lcModel-method), 21
externalMetric, lcModel, lcModel-method, 21
externalMetric, lcModels, lcModel-method
(externalMetric, lcModel, lcModel-method), 21
externalMetric, lcModels, lcModels-method
(externalMetric, lcModel, lcModel-method), 21
externalMetric, lcModels, lcModels-method
(externalMetric, lcModel, lcModel-method), 21
externalMetric, list, lcModel-method
(externalMetric, lcModel, lcModel-method), 21

fit(lcMethod-class), 36
fit, lcMethod-method (lcMethod-class), 36

fitted.lcApproxModel
(lcApproxModel-class), 35
fitted.lcModel, 12–14, 19, 22, 64, 68, 72, 73, 78, 79, 81, 82, 85, 86, 88, 90

flexmix::flexmix, 44, 45
flexmix::FLXMRglm, 45

formula.lcModel, 6–9, 21, 23, 38, 94, 96
formula.lcModel, 24

funFEM::funFEM, 46

generateLongData, 24, 33, 34
generateMetricDefinition, 18, 22, 25, 26, 27, 70
generateMetricNames, 18, 22, 26, 27, 70

INDEX

99
getInternalMetricDefinition, 18, 22, 26, 26, 27, 70
getInternalMetricNames, 18, 22, 26, 27, 70
g etLabel, lcMethod-method, 36
getLabel, lcMethod-method (lcMethod-class), 36
getLabel, lcModel-method (lcModel-class), 62
getLcMethod, 27
g etName, lcMethod-method (lcMethod-class), 36
g etName, lcMethod-method (lcMethod-class), 36
g etShortName, lcMethod-method (lcMethod-class), 36
g etShortName, lcModel-method (lcModel-class), 36
g etShortName, lcModel-method (lcModel-class), 62
idVariable, lcMethod-method (idVariable), 28
idVariable, lcModel-method (idVariable), 28
kml::kml, 48
kml::parALGO, 48
latrend, 29, 31–36, 95
latrend-package, 5
latrendBatch, 30, 30, 32, 33, 35
latrendBoot, 15, 16, 30, 31, 31, 33, 35
latrendCV, 15, 16, 30–32, 32, 35
latrendData, 33
latrendRep, 30–33, 34
lcApproxModel (lcApproxModel-class), 35
lcApproxModel-class, 35
lcMethod, 36, 39
lcMethod-class, 36, 36, 63
lcMethod.call, 36, 39
lcMethodAKMedoids, 38, 40, 41–43, 46–49, 51–56, 58, 61, 62
lcMethodCrimCV, 38, 40, 41, 42, 43, 46–49, 51–56, 58, 61, 62
lcMethodCustom, 38, 40, 41, 42, 43, 46–49, 51–56, 58, 61, 62
lcMethodDtwclust, 38, 40–42, 43, 46–49, 51–56, 58, 61, 62
lcMethodFlexmix, 44, 45
lcMethodFlexmixGBTM, 44, 45
lcMethodFunFEM, 38, 40–43, 46, 47–49, 51–56, 58, 61, 62
lcMethodGCKM, 38, 40–43, 46, 47, 48, 49, 51–56, 58, 61, 62
lcMethodKML, 38, 40–43, 46, 47, 48, 49, 51–56, 58, 61, 62
lcMethodLcmmGBTM, 38, 40–43, 46–48, 49, 51–56, 58, 61, 62
lcMethodLcmmGMM, 38, 40–43, 46–49, 50, 52–56, 58, 61, 62
lcMethodLMKM, 38, 40–43, 46–49, 51, 51, 52–56, 58, 61, 62
lcMethodLongclust, 38, 40–43, 46–49, 51, 52, 52–56, 58, 61, 62
lcMethodMclustLLPA, 38, 40–43, 46–49, 51, 52, 53–56, 58, 61, 62
lcMethodMixAK_GLM, 38, 40–43, 46–49, 51–53, 54, 55, 56, 58, 61, 62
lcMethodMixtoolsGMM, 38, 40–43, 46–49, 51–54, 55, 56, 58, 61, 62
lcMethodMixtoolsNPRM, 38, 40–43, 46–49, 51–55, 56, 58, 61, 62
lcMethodMixTVEM, 57
lcMethodRandom, 38, 40–43, 46–49, 51–56, 58, 61, 62
lcMethods, 59
lcMethodStratify, 38, 40–43, 46–49, 51–56, 58, 60, 62
lcMethodTwoStep, 38, 40–43, 46–49, 51–56, 58, 61, 61
lme4::lmer, 12–14, 19, 22, 64, 67, 72, 73, 78, 79, 81, 82, 85, 86, 88, 90
longclust::longclustEM, 52
max.lcModels, 68, 70, 71
mclust::Mclust, 53
meltRepeatedMeasures, 69
metric, 18, 22, 26, 27, 70, 84
metric, lcModel-method (metric), 70
metric, lcModels-method (metric), 70
metric, list-method (metric), 70
min.lcModels. 68, 70, 71
mixAK::GLMM_MCMC, 54
mixtools::npEM, 56
mixtools::regmixEM.mixed, 55
model.data.lcModel, 66, 71
model.frame.lcModel, 12–14, 19, 22, 64, 68,
72, 73, 78, 79, 81, 82, 85, 86, 88, 90

names, lcMethod-method (lcMethod-class), 36
nClusters, 72
nIds, 73
nobs.lcModel, 12–14, 19, 22, 64, 68, 72, 73,
78, 79, 81, 82, 85, 86, 88, 90

plot, lcModel, ANY-method, 74
plotClusterTrajectories, 74, 74, 84
plotClusterTrajectories, data.frame-method
(plotClusterTrajectories), 74
plotClusterTrajectories, lcModel-method
(plotClusterTrajectories), 74
plotMetric, 76
plotTrajectories, 77
plotTrajectories, data.frame-method
(plotTrajectories), 77
plotTrajectories, lcModel-method
(plotTrajectories), 77
postFit, lcMethod-method
(lcMethod-class), 36
postFit, lcMethod-method
(lcMethod-class), 36
postprob, 12–14, 19, 22, 64, 68, 72, 73, 78,
79, 81, 82, 85, 86, 88, 90
postprob, lcModel-method (postprob), 78
postprobFromAssignments, 78
predict.lcApproxModel
(lcApproxModel-class), 35
predict.lcModel, 12–14, 19, 22, 64, 68, 72,
73, 78, 79, 81, 82, 85, 86, 88, 90
predictAssignments, 12–14, 19, 22, 64, 68,
72, 73, 78, 79, 80, 81, 82, 85, 86, 88, 90
predictAssignments, lcModel-method
(predictAssignments), 80
predictForCluster, 12–14, 19, 22, 64, 68,
72, 73, 78, 79, 81, 82, 85, 86, 88, 90
predictForCluster, lcModel-method
(predictForCluster), 81
predictPostprob, 12–14, 19, 22, 64, 68, 72,
73, 78–81, 82, 85, 86, 88, 90
predictPostprob, lcModel-method
(predictPostprob), 82
preFit (lcMethod-class), 36
preFit, lcMethod-method
(lcMethod-class), 36
prepareData, lcModel-method
(lcMethod-class), 36
print.lcMethod, 82
print.lcModels, 8, 66, 83, 87
qqPlot, 84
qqPlot, lcModel-method (qqPlot), 84
R.utils::Verbose, 29, 30, 32–34, 37, 95
residuals.lcModel, 12–14, 19, 22, 64, 68,
72, 73, 78, 79, 81, 82, 84, 86, 88, 90
responseVariable, 29, 85, 89
responseVariable, lcModel-method
(responseVariable), 85
responseVariable, lcModel-method
(responseVariable), 85
sigma.lcModel, 12–14, 19, 22, 64, 68, 72, 73,
78, 79, 81, 82, 85, 86, 88, 90
stats::lm, 51
strip, 86
strip, lcModel-method (strip), 86
strip, lcModel-method (strip), 86
subset.lcModels, 8, 66, 83, 87
summary.lcModel, 88
time.lcModel, 12–14, 19, 22, 64, 68, 72, 73,
78, 79, 81, 82, 85, 86, 88, 90
timeVariable, 29, 85, 89
timeVariable, lcModel-method
(timeVariable), 89
timeVariable, lcModel-method
(timeVariable), 89
trajectories, 12–14, 19, 22, 64, 68, 72, 73,
77–79, 81, 82, 85, 86, 88, 89

INDEX
trajectories, lcModel-method
  (trajectories), 89
trajectoryAssignments, 14, 75, 91, 95
trajectoryAssignments, lcModel-method
  (trajectoryAssignments), 91
transformFitted, 91
transformFitted, data.frame, lcModel-method
  (transformFitted), 91
transformFitted, list, lcModel-method
  (transformFitted), 91
transformFitted, matrix, lcModel-method
  (transformFitted), 91
transformFitted, NULL, lcModel-method
  (transformFitted), 91
transformLatrendData, 92
transformLatrendData, call-method
  (transformLatrendData), 92
transformLatrendData, data.frame-method
  (transformLatrendData), 92
transformLatrendData, matrix-method
  (transformLatrendData), 92
transformPredict, 93
transformPredict, data.frame, lcModel-method
  (transformPredict), 93
transformPredict, matrix, lcModel-method
  (transformPredict), 93
transformPredict, NULL, lcModel-method
  (transformPredict), 93
transformPredict, vector, lcModel-method
  (transformPredict), 93
update.lcMethod, 6–9, 21, 23, 38, 94, 96
update.lcModel, 95
validate (lcMethod-class), 36
validate, lcMethod-method
  (lcMethod-class), 36
Verbose, 29, 30, 32–34, 95
which.is.max, 91
which.max, 91
which.weight, 14, 91, 95