Package 'em'

January 11, 2023

Title Generic EM Algorithm

Version 1.0.0

Description A generic function for running the Expectation-Maximization (EM) algorithm within a maximum likelihood framework, based on Dempster, Laird, and Rubin (1977) <doi:10.1111/j.2517-6161.1977.tb01600.x> is implemented. It can be applied after a model fitting using R's existing functions and packages.

URL https://github.com/wudongjie/em

License GPL (>= 3)

Encoding UTF-8

SystemRequirements C++11

BugReports https://github.com/wudongjie/em/issues

NeedsCompilation yes

Depends R (>= 3.0.0)

Imports stats, utils, survival, plm, methods, mclust, dplyr, numDeriv, nnet, magrittr

LinkingTo Rcpp, RcppArmadillo

Suggests testthat (>= 3.0.0), parallel, fitdistrplus, gnm

Config/testthat/edition 3

RoxygenNote 7.2.3

LazyData true

Author Dongjie Wu [aut, cre, cph] (<https://orcid.org/0000-0002-4490-8657>)

Maintainer Dongjie Wu <dwu.jacob@gmail.com>

Repository CRAN

Date/Publication 2023-01-11 06:20:02 UTC

R topics documented:

cstep *C-Step of EM algorithm*

Description

Given the posterior probability, generate a matrix to assign each individual to a class. The assignment is based on which probability is the largest.

Usage

```
cstep(postpr)
```

Arguments

postpr	(‘matrix()‘) The matrix of the posterior probability

em *A Generic EM Algorithm*

Description

This is a generic EM algorithm that can work on specific models/objects. Currently, it supports ‘lm‘, ‘glm‘, ‘gnm‘ in package gnm, ‘clogit‘ in package survival and ‘multinom‘ in package nnet. Use ‘?em.default‘ to check the manual of the default function of ‘em‘.

Usage

```
em(object, ...)
```

Arguments

object	the model used, e.g. ‘lm‘, ‘glm‘, ‘gnm‘, ‘clogit‘, ‘multinom‘
...	arguments used in the ‘model‘.

Value

An object of class ‘em‘ is a list containing at least the following components: `models` a list of models/objects whose class are determined by a model fitting from the previous step. `pi` the prior probabilities. `latent` number of the latent classes. `algorithm` the algorithm used (could be either ‘em‘, ‘sem‘ or ‘cem‘). `obs` the number of observations. `post_pr` the posterior probabilities. `concomitant` a list of the concomitant model. It is empty if no concomitant model is used. `init.method` the initialization method used. `call` the matched call. `terms` the codeterms object used.

Author(s)

Dongjie Wu

Examples

```
fit.lm <- lm(yn ~ x, data = simreg)
results <- em(fit.lm, latent = 2, verbose = FALSE)
fmm_fit <- predict(results)
fmm_fit_post <- predict(results, prob = "posterior")
```

`em.clogit` *The em function for 'survival::clogit'.*

Description

The em function for 'survival::clogit'.

Usage

```
## S3 method for class 'clogit'
em(
  object,
  latent = 2,
  verbose = FALSE,
  init.method = c("random", "kmeans", "hc"),
  init.prob = NULL,
  algo = c("em", "cem", "sem"),
  cluster.by = NULL,
  max_iter = 500,
  abs_tol = 1e-04,
  concomitant = list(...),
  use.optim = FALSE,
  optim.start = c("random", "sample5"),
  ...
)
```

Arguments

`object`	the model used, e.g. 'lm', 'glm', 'gnm'.
`latent`	the number of latent classes.
`verbose`	'True' to print the process of convergence.
`init.method`	the initialization method used in the model. The default method is 'random'. 'kmeans' is K-means clustering. 'hc' is model-based agglomerative hierarchical clustering.

`init.prob`	the starting prior probabilities used in classification based method.
`algo`	the algorithm used in em: 'em' the default EM algorithm, the classification em 'cem', or the stochastic em 'sem'.
`cluster.by`	a variable to define the level of clustering.
`max_iter`	the maximum iteration for em algorithm.
`abs_tol`	absolute accuracy requested.
`concomitant`	the formula to define the concomitant part of the model. The default is NULL.
`use.optim`	maximize the complete log likelihood (MLE) by using 'optim' and 'rcpp' code.The default value is 'FALSE'.
`optim.start`	the initialization method of generating the starting value for MLE.
`...`	arguments used in the 'model'.

Value

An object of class 'em' is a list containing at least the following components: `models` a list of models/objects whose class are determined by a model fitting from the previous step. `pi` the prior probabilities. `latent` number of the latent classes. `algorithm` the algorithm used (could be either 'em', 'sem' or 'cem'). `obs` the number of observations. `post_pr` the posterior probabilities. `concomitant` a list of the concomitant model. It is empty if no concomitant model is used. `init.method` the initialization method used. `call` the matched call. `terms` the codeterms object used.

`em.default` *The default em function*

Description

The default em function

Usage

```
## Default S3 method:
em(
  object,
  latent = 2,
  verbose = FALSE,
  init.method = c("random", "kmeans", "hc"),
  init.prob = NULL,
  algo = c("em", "cem", "sem"),
  cluster.by = NULL,
  max_iter = 500,
  abs_tol = 1e-04,
  concomitant = list(...),
  use.optim = FALSE,
  optim.start = c("random", "sample5"),
  ...
)
```

Arguments

object	the model used, e.g. ‘lm‘, ‘glm‘, ‘gnm‘.
latent	the number of latent classes.
verbose	‘True‘ to print the process of convergence.
init.method	the initialization method used in the model. The default method is ‘random‘. ‘kmeans‘ is K-means clustering. ‘hc‘ is model-based agglomerative hierarchical clustering.
init.prob	the starting prior probabilities used in classification based method.
algo	the algorithm used in em: ‘em‘ the default EM algorithm, the classification em ‘cem‘, or the stochastic em ‘sem‘.
cluster.by	a variable to define the level of clustering.
max_iter	the maximum iteration for em algorithm.
abs_tol	absolute accuracy requested.
concomitant	the formula to define the concomitant part of the model. The default is NULL.
use.optim	maximize the complete log likelihood (MLE) by using ‘optim‘ and ‘rcpp‘ code.The default value is ‘FALSE‘.
optim.start	the initialization method of generating the starting value for MLE.
...	arguments used in the ‘model‘.

Value

An object of class ‘em‘ is a list containing at least the following components: `models` a list of models/objects whose class are determined by a model fitting from the previous step. `pi` the prior probabilities. `latent` number of the latent classes. `algorithm` the algorithm used (could be either ‘em‘, ‘sem‘ or ‘cem‘). `obs` the number of observations. `post_pr` the posterior probabilities. `concomitant` a list of the concomitant model. It is empty if no concomitant model is used. `init.method` the initialization method used. `call` the matched call. `terms` the codeterms object used.

`em.fitdist` *The default em function*

Description

The default em function

Usage

```
## S3 method for class 'fitdist'
em(
  object,
  latent = 2,
  verbose = FALSE,
```

```
  init.method = c("random", "kmeans", "hc"),
  init.prob = NULL,
  algo = c("em", "cem", "sem"),
  max_iter = 500,
  ...
)
```

Arguments

object	the model used, e.g. 'lm', 'glm', 'gnm'.
latent	the number of latent classes.
verbose	'True' to print the process of convergence.
init.method	the initialization method used in the model. The default method is 'random'. 'kmeans' is K-means clustering. 'hc' is model-based agglomerative hierarchical clustering.
init.prob	the starting prior probabilities used in classification based method.
algo	the algorithm used in em: 'em' the default EM algorithm, the classification em 'cem', or the stochastic em 'sem'.
max_iter	the maximum iteration for em algorithm.
...	arguments used in the 'model'.

Value

An object of class 'em' is a list containing at least the following components: `models` a list of models/objects whose class are determined by a model fitting from the previous step. `pi` the prior probabilities. `latent` number of the latent classes. `algorithm` the algorithm used (could be either 'em', 'sem' or 'cem'). `obs` the number of observations. `post_pr` the posterior probabilities. `concomitant` a list of the concomitant model. It is empty if no concomitant model is used. `init.method` the initialization method used. `call` the matched call. `terms` the codeterms object used.

---

em.glmerMod *The em function for glmerMod*

Description

The em function for glmerMod

Usage

```
## S3 method for class 'glmerMod'
em(
  object,
  latent = 2,
  verbose = FALSE,
```

```
  init.method = c("random", "kmeans", "hc"),
  algo = c("em", "cem", "sem"),
  max_iter = 500,
  concomitant = list(...),
  ...
)
```

Arguments

object	the model used, e.g. 'lm', 'glm', 'gnm'.
latent	the number of latent classes.
verbose	'True' to print the process of convergence.
init.method	the initialization method used in the model. The default method is 'random'. 'kmeans' is K-means clustering. 'hc' is model-based agglomerative hierarchical clustering.
algo	the algorithm used in em: 'em' the default EM algorithm, the classification em 'cem', or the stochastic em 'sem'.
max_iter	the maximum iteration for em algorithm.
concomitant	the formula to define the concomitant part of the model. The default is NULL.
...	arguments used in the 'model'.

Value

An object of class 'em' is a list containing at least the following components: `models` a list of models/objects whose class are determined by a model fitting from the previous step. `pi` the prior probabilities. `latent` number of the latent classes. `algorithm` the algorithm used (could be either 'em', 'sem' or 'cem'). `obs` the number of observations. `post_pr` the posterior probabilities. `concomitant` a list of the concomitant model. It is empty if no concomitant model is used. `init.method` the initialization method used. `call` the matched call. `terms` the codeterms object used.

`em.panelmodel` *The em function for 'panelmodel' such as 'plm'.*

Description

The em function for 'panelmodel' such as 'plm'.

Usage

```
## S3 method for class 'panelmodel'
em(
  object,
  latent = 2,
  verbose = FALSE,
```

```
  init.method = c("random", "kmeans"),
  algo = c("em", "cem", "sem"),
  max_iter = 500,
  concomitant = list(...),
  ...
)
```

Arguments

object	the model used, e.g. 'lm', 'glm', 'gnm', 'plm'.
latent	the number of latent classes.
verbose	'True' to print the process of convergence.
init.method	the initialization method used in the model. The default method is 'random'.
algo	the algorithm used in em: the default EM algorithm, the classification em 'cem', or the stochastic em 'sem'.
max_iter	the maximum iteration for em algorithm.
concomitant	the formula to define the concomitant part of the model. The default is NULL.
...	arguments used in the 'model'.

Value

An object of class 'em' is a list containing at least the following components: `models` a list of models/objects whose class are determined by a model fitting from the previous step. `pi` the prior probabilities. `latent` number of the latent classes. `algorithm` the algorithm used (could be either 'em', 'sem' or 'cem'). `obs` the number of observations. `post_pr` the posterior probabilities. `concomitant` a list of the concomitant model. It is empty if no concomitant model is used. `init.method` the initialization method used. `call` the matched call. `terms` the codeterms object used.

---

estep *This function performs an E-Step of EM Algorithm.*

Description

This function performs an E-Step of EM Algorithm.

Usage

```
estep(models, pi_matrix)
```

Arguments

models	models used in the EM algorithm,
pi_matrix	the pi matrix.

Value

the fitting result for the model.

`fit.den` *Fit the density function for a fitted model.*

Description

This function generates the probability density of given models.

Usage

```
fit.den(object, ...)
```

Arguments

`object`	the fitted model such as ‘lm‘.
`...`	other used arguments.

Value

the density function.

`fit.den.coxph` *Fit the density for the survival::clogit*

Description

Fit the density for the survival::clogit

Usage

```
## S3 method for class 'coxph'
fit.den(object, ...)
```

Arguments

`object`	the fitted model.
`...`	other used arguments.

Value

the density function.

fit.den.fitdist *Fitting the density function using in 'fitdistrplus::fitdist()'*

Description

Fitting the density function using in 'fitdistrplus::fitdist()'

Usage

```
## S3 method for class 'fitdist'
fit.den(object, ...)
```

Arguments

object	the fitted model.
...	other used arguments.

Value

the density function.

fit.den.glm *Fit the density function for a generalized linear regression model.*

Description

Fit the density function for a generalized linear regression model.

Usage

```
## S3 method for class 'glm'
fit.den(object, ...)
```

Arguments

object	the fitted model.
...	other used arguments.

Value

the density function.

fit.den.glmerMod *Fit the density function for a generalized linear mixed effect model.*

Description

Fit the density function for a generalized linear mixed effect model.

Usage

```
## S3 method for class 'glmerMod'
fit.den(object, ...)
```

Arguments

object	the fitted model.
...	other used arguments.

Value

the density function.

fit.den.gnm *Fit the density function for a generalized non-linear regression model.*

Description

Fit the density function for a generalized non-linear regression model.

Usage

```
## S3 method for class 'gnm'
fit.den(object, ...)
```

Arguments

object	the fitted model.
...	other used arguments.

Value

the density function.

fit.den.lm — *Fit the density function for a linear regression model.*

Description

Fit the density function for a linear regression model.

Usage

```
## S3 method for class 'lm'
fit.den(object, ...)
```

Arguments

object	the fitted model.
...	other used arguments.

Value

the density function.

fit.den.multinom — *Fit the density function for a multinomial regression model.*

Description

Fit the density function for a multinomial regression model.

Usage

```
## S3 method for class 'multinom'
fit.den(object, ...)
```

Arguments

object	the fitted model.
...	other used arguments.

Value

the density function.

`fit.den.nnet` *Fit the density function for a 'nnet' model.*

Description

Fit the density function for a 'nnet' model.

Usage

```
## S3 method for class 'nnet'
fit.den(object, ...)
```

Arguments

`object`	the fitted model.
`...`	other used arguments.

Value

the density function.

`fit.den.plm` *Fit the density function for a panel regression model.*

Description

Fit the density function for a panel regression model.

Usage

```
## S3 method for class 'plm'
fit.den(object, ...)
```

Arguments

`object`	the fitted model.
`...`	other used arguments.

Value

the density function.

`flatten` *Flatten a data.frame or matrix by column or row with its name. The name will be transformed into the number of row/column plus the name of column/row separated by '.'.*

Description

Flatten a data.frame or matrix by column or row with its name. The name will be transformed into the number of row/column plus the name of column/row separated by '.'.

Usage

```
flatten(x, by = c("col", "row"))
```

Arguments

`x`	a data.frame or matrix.
`by`	either by column or by row.

Value

a flattened vector with names

`init.em` *Initialization of EM algorithm*

Description

Given a matrix with number of rows equal to the number of observation and number of columns equal to the number of latent classes, function 'init.em' generate the posterior probability using that matrix based on the method set by the user.

Usage

```
init.em(object, ...)
```

Arguments

`object`	A matrix.
`...`	other used arguments.

Value

The posterior probability matrix

init.em.hc *model-based agglomerative hierarchical clustering*

Description

model-based agglomerative hierarchical clustering

Usage

```
## S3 method for class 'hc'
init.em(object, ...)
```

Arguments

object	A matrix.
...	other used arguments.

Value

The posterior probability matrix

init.em.kmeans *K-mean initialization*

Description

K-mean initialization

Usage

```
## S3 method for class 'kmeans'
init.em(object, ...)
```

Arguments

object	A matrix.
...	other used arguments.

Value

The posterior probability matrix

init.em.random *Random initialization*

Description

Random initialization

Usage

```
## S3 method for class 'random'
init.em(object, ...)
```

Arguments

object	A matrix.
...	other used arguments.

Value

The posterior probability matrix

init.em.random.weights *Random initialization with weights*

Description

Random initialization with weights

Usage

```
## S3 method for class 'random.weights'
init.em(object, ...)
```

Arguments

object	A matrix.
...	other used arguments.

Value

The posterior probability matrix

init.em.sample5 *Initialization using sampling 5 times.*

Description

Initialization using sampling 5 times.

Usage

```
## S3 method for class 'sample5'
init.em(object, ...)
```

Arguments

object	A matrix.
...	other used arguments.

Value

The posterior probability matrix

logLik.em *This function computes logLik of EM Algorithm.*

Description

This function computes logLik of EM Algorithm.

Usage

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

Arguments

object	an object of ‘em‘.
...	other used arguments.

Value

the log-likelihood value

mstep *M-Step of EM algorithm*

Description

This function performs an M-Step of EM Algorithm.

Usage

```
mstep(models, post_pr = NULL)
```

Arguments

models	the models used in the EM algorithm
post_pr	the posterior probability.

Value

the fitting result for the model.

mstep.concomitant *The mstep for the concomitant model.*

Description

This section was inspired by Flexmix.

Usage

```
mstep.concomitant(formula, data, postpr)
```

Arguments

formula	the formula of the concomitant model.
data	the data or model.frame related to the concomitant model.
postpr	the posterior probability matrix.

Value

the function returns a fitted nnet object.

`mstep.concomitant.refit` *The refit of for the concomitant model. This section was inspired by Flexmix.*

Description

The refit of for the concomitant model. This section was inspired by Flexmix.

Usage

```
mstep.concomitant.refit(formula, data, postpr)
```

Arguments

`formula`	the formula of the concomitant model.
`data`	the data or model.frame related to the concomitant model.
`postpr`	the posterior probability matrix.

Value

the function returns a fitted multinom object.

`multi.em` *Multiple run of EM algorithm*

Description

Multiple run of EM algorithm

Usage

```
multi.em(object, ...)
```

Arguments

`object`	the model to use in em, e.g. 'lm', 'glm', 'gnm'
`...`	arguments used in em.

Value

return the 'em' object with the maximum log-likelihood.

multi.em.default *Default generic for multi.em*

Description

Default generic for multi.em

Usage

```
## Default S3 method:
multi.em(
  object,
  iter = 10,
  parallel = FALSE,
  num.cores = 2,
  random.init = TRUE,
  ...
)
```

Arguments

object	the model to use in em, e.g. 'lm', 'glm', 'gnm'
iter	number of iterations for running EM algorithm.
parallel	whether to use the parallel computing.
num.cores	number of cores used in the parallel computing.
random.init	whether to use a random initialization.
...	arguments used in em.

Value

return the 'em' object with the maximum log-likelihood.

plot.em *Plot the fitted results of EM algorithm*

Description

This is the generic plot function for 'em' project. One can produce three types of graphs using this function 1. A graph of the predicted value distribution for each component. 2. A histogram of posterior probability distributions

Usage

```
## S3 method for class 'em'
plot(
  x,
  by = c("component", "prob"),
  prior = FALSE,
  cols = rep(1, length(x$models)),
  lwds = rep(3, length(x$models)),
  ltys = c(seq_len(length(x$models))),
  ranges = NULL,
  main = NULL,
  lgd = list(),
  lgd.loc = "topleft",
  hist.args = list(main = "Histograms of posterior probabilities", xlab =
    "Posterior Probabilities"),
  ...
)
```

Arguments

x	the ‘em‘ model to plot
by	the type of the graph to produce. The default is ‘component‘.
prior	whether fit the model using prior probabilities.
cols	lines’ colors.
lwds	Lines’ widths.
ltys	lines’ types.
ranges	the ranges of the x-axis and the y-axis limits of plots. It should be a vector of four numeric values. The first two represent the x-axis limits. The last two represent the y-axis limits
main	the main title.
lgd	a list for legend related arguments.
lgd.loc	the location of the legend. The default is "topleft".
hist.args	The list of arguments for the histogram.
...	other arguments.

Value

‘NULL‘

predict.em *Predict the fitted finite mixture models*

Description

Predict the fitted finite mixture models

Usage

```
## S3 method for class 'em'
predict(object, prob = c("prior", "posterior"), ...)
```

Arguments

object	Output from em, representing a fitted model using EM algorithm.
prob	the probabilities used to compute the fitted value. It can be either prior probability (‘prior‘) or posterior probability (‘posterior‘). The default value is ‘prior‘.
...	other arguments.

Value

An object of class ‘predict.em‘ is a list containing at least the following components: components a list of fitted values by components with each element a matrix/vector of fitted values. mean a matrix of predicted values computed by weighted sum of fitted values by components. The weights used in the computation can be either prior probabilities or posterior probabilities depending on the parameter ‘prob‘. prob the value used in the parameter ‘prob‘.

print.em *Print the ‘em‘ object*

Description

Print the ‘em‘ object

Usage

```
## S3 method for class 'em'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

x	the ‘em‘ object.
digits	the maximum digits printed, the default is ‘3L‘.
...	other arguments used.

Value

print the ‘em‘ object on the screen.

`print.summary.em` *Print the ‘summary.em‘ object*

Description

Print the ‘summary.em‘ object

Usage

```
## S3 method for class 'summary.em'
print(
  x,
  digits = max(3L, getOption("digits") - 3L),
  signif.stars = getOption("show.signif.stars"),
  ...
)
```

Arguments

`x`	the ‘summary.em‘ object.
`digits`	the maximum digits printed, the default is ‘3L‘.
`signif.stars`	logical; if ‘TRUE‘, P-values are additionally encoded visually as ‘significance stars‘ in order to help scanning of long coefficient tables. It defaults to the ‘show.signif.stars‘ slot of options.
`...`	other augments used in ‘printCoefmat‘.

Value

print the ‘summary.em‘ object on the screen.

`simbinom` *Simulated Data from a logistic regression*

Description

A data set with simulated data from a mixture of a logistic regression.

Usage

```
simbinom
```

Format

A data frame with 10000 rows and 2 variables:

y A dependent variable generated from a mixture of a logistic regression with x

x An independent variable

Source

<https://www.github.com/wudongjie/em>

`simclogit` *Simulated Data from a conditional logistic regression*

Description

A data set with simulated data from a mixture of a conditional logistic regression.

Usage

```
simclogit
```

Format

A data frame with 10000 rows and 4 variables:

x2 A dummy variable showing whether x is equal to level 2

x3 A dummy variable showing whether x is equal to level 3

a2 Whether the alternative choice 2 is chosen

a2_x2 Interaction between a2 and x2

a2_x3 Interaction between a2 and x3

a3 Whether the alternative choice 3 is chosen

a3_x2 Interaction between a3 and x2

a3_x3 Interaction between a3 and x3

chosen1 Whether the observation-alternative combination is chosen (Generated by a one-class regression).

chosen2 Whether the observation-alternative combination is chosen (Generated by a two-class mixed regression).

fid Family ID

id Individual ID

z Other variables

Source

<https://www.github.com/wudongjie/em>

simreg *Simulated Regression Data*

Description

A data set with simulated data from mixture regression models.

Usage

```
simreg
```

Format

A data frame with 1000 rows and 5 variables:

yp A dependent variable generated from a mixture of a poisson regression with x

yn A dependent variable generated from a mixture of a linear regression with x

yc A dependent variable generated from a mixture of a linear regression with x and a concomitant variable of z

x An independent variable

z A concomitant variable

Source

<https://www.github.com/wudongjie/em>

sstep *S-step of EM algorithm*

Description

Given the posterior probability, generate a matrix to assign each individual to a class. The assignment is randomly sampled based on the posterior probability.

Usage

```
sstep(postpr)
```

Arguments

postpr	(‘matrix()‘) The matrix of the posterior probability

summary.em *Summaries of fitted finite mixture models using EM algorithm*

Description

Summaries of fitted finite mixture models using EM algorithm

Usage

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

Arguments

object	Output from em, representing a fitted model using EM algorithm.
...	other arguments used.

Value

An object of class 'summary.em' is a list containing at least the following components: call the matched call. coefficients pi the prior probabilities. latent number of the latent classes. ll log-likelihood value. sum.models summaries of models generated by 'summary()' of models from each class. df degree of freedom. obs number of observations. AIC the Akaike information criterion. BIC the Bayesian information criterion. concomitant a list of the concomitant model. It is empty if no concomitant model is used. concomitant.summary summaries of the concomitant model generated by 'summary()'.

vdummy *Transform a factor variable to a matrix of dummy variables*

Description

Transform a factor variable to a matrix of dummy variables

Usage

```
vdummy(x)
```

Arguments

x	a factor vector

Value

a matrix of dummy variables

Index