Package ‘EMMIXmfa’

December 16, 2019

Type Package

Title Mixture Models with Component-Wise Factor Analyzers

Version 2.0.11

Date 2019-12-16

URL https://github.com/suren-rathnayake/EMMIXmfa

Author Suren Rathnayake, Geoff McLachlan, David Peel, Jangsun Baek

Maintainer Suren Rathnayake <surenr@gmail.com>

Description We provide functions to fit finite mixtures of multivariate normal or t-distributions to data with various factor analytic structures adopted for the covariance/scale matrices. The factor analytic structures available include mixtures of factor analyzers and mixtures of common factor analyzers. The latter approach is so termed because the matrix of factor loadings is common to components before the component-specific rotation of the component factors to make them white noise. Note that the component-factor loadings are not common after this rotation. Maximum likelihood estimators of model parameters are obtained via the Expectation-Maximization algorithm. See descriptions of the algorithms used in
McLachlan GJ, Peel D (2000) <doi:10.1002/0471721182.ch8>
McLachlan GJ, Peel D (2000) <ISBN:1-55860-707-2>
McLachlan GJ, Peel D, Bean RW (2003) <doi:10.1016/S0167-9473(02)00183-4>
McLachlan GJ, Bean RW, Ben-Tovim Jones L (2007) <doi:10.1016/j.csda.2006.09.015>
Baek J, McLachlan GJ, Flack LK (2010) <doi:10.1109/TPAMI.2009.149>
Baek J, McLachlan GJ (2011) <doi:10.1093/bioinformatics/btr112>
McLachlan GJ, Baek J, Rathnayake SI (2011) <doi:10.1002/9781119995678.ch9>.

Suggests mvtnorm, GGally, ggplot2

License GPL (>= 2)

NeedsCompilation yes

Repository CRAN

Date/Publication 2019-12-16 22:50:02 UTC

R topics documented:

`EMMIXmfa-package` *Mixture Models with Component-Wise Factor Analyzers*

Description

This package provides functions for fitting mixtures of factor analyzers (MFA) and mixtures of common factor analyzers (MCFA) models.

MFA and MCFA models belong to the class of finite mixture models, that adopt factor models for the component-covariance matrices. More specifically, under the factor model, the correlations between feature variables can be explained by the linear dependance of these variables on a smaller small number q of (unobservable) latent factors. The component distributions can be either from the family of multivariate normals or from the family of multivariate t-distributions. Maximum likelihood estimation of the model parameters is implemented using the Expectation–Maximization algorithm.

The joint distribution of the factors and errors can be taken to be either the multivariate normal or t-distribution. The factor analytic representation of the component-covariance matrices is a way of dimension reduction in that it enables the mixture distributions to be fitted to data with dimension p relatively large compared to the sample size n.

Unlike MFA, MCFA models can be used to display the observed data points in the q-dimensional factor space. The MCFA would also provide a greater reduction in the number of parameters in the model.

Details

Package:	EMMIXmfa
Type:	Package
Version:	2.0.4
Date:	2018-09-17
License:	GPL (>= 2)

Author(s)

Suren Rathnayake, Geoffrey McLachlan, David Peel, Jangsun Baek

References

Baek J, and McLachlan GJ (2008). Mixtures of factor analyzers with common factor loadings for the clustering and visualisation of high-dimensional data. *Technical Report NI08018-SCH*, Preprint Series of the Isaac Newton Institute for Mathematical Sciences, Cambridge.

Baek J, McLachlan GJ, and Flack LK (2010). Mixtures of factor analyzers with common factor loadings: applications to the clustering and visualisation of high-dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **32**, 2089–2097.

Baek J, and McLachlan GJ (2011). Mixtures of common *t*-factor analyzers for clustering highdimensional microarray data. *Bioinformatics* **27**, 1269–1276.

McLachlan GJ, Baek J, and Rathnayake SI (2011). Mixtures of factor analyzers for the analysis of high-dimensional data. In *Mixture Estimation and Applications*, KL Mengersen, CP Robert, and DM Titterington (Eds). Hoboken, New Jersey: Wiley, pp. 171–191.

McLachlan GJ and Peel D (2000). *Finite Mixture Models*. New York: Wiley.

McLachlan GJ, and Peel D (2000). Mixtures of factor analyzers. In *Proceedings of the Seventeenth International Conference on Machine Learning*, P. Langley (Ed.). San Francisco: Morgan Kaufmann, pp. 599–606.

McLachlan GJ, Bean RW, Ben-Tovim Jones L (2007). Extension of the mixture of factor analyzers model to incorporate the multivariate *t* distribution. *Computational Statistics & Data Analysis*, **51**, 5327–5338.

McLachlan GJ, Peel D, and Bean RW (2003). Modelling high-dimensional data by mixtures of factor analyzers. *Computational Statistics & Data Analysis* **41**, 379–388.

Examples

```
set.seed(1)
Y <- iris[, -5]
mfa_model <- mfa(Y, g = 3, q = 3)
mtfa_model <- mtfa(Y, g = 3, q = 3)
mcfa_model <- mcfa(Y, g = 3, q = 3)
mctfa_model <- mctfa(Y, g = 3, q = 3)
```

`ari` *Computes adjusted Rand Index*

Description

Computes adjusted Rand index.

Usage

```
ari(cls, hat_cls)
```

Arguments

cls	A numeric or character vector of labels.
hat_cls	A numeric or character vector of labels same length as cls.

Details

Measures the agreement between two sets of partitions. The upper bound of 1 implies perfect agreement. The expected value is zero if the partitions are random.

Value

Scaler specifying how closely two partitions agree.

References

Hubert L, and Arabie P (1985). Comparing Partitions. *Journal of the Classification* **2**, 193–218.

See Also

minmis

Examples

```
set.seed(1984)
Y <- scale(iris[, -5])
model <- mfa(Y, g = 3, q = 3, nkmeans = 1, nrandom = 0)
#
ari(model$clust, iris[, 5])
#
minmis(model$clust, iris[, 5])
```

---

factor_scores *Computes Factor Scores*

Description

This function computes factor scores for observations. Using factor scores, we can represent the original data point y_j in a q-dimensional reduced space. This is only meaningful in the case of mcfa or mctfa models, as the factor cores for mfa and mtfa are white noise.

The (estimated conditional expectation of) unobservable factors U_{ij} given y_j and the component membership can be expressed by,

$$\hat{u}_{ij} = E_{\hat{\Psi}}\{U_{ij} \mid y_j, z_{ij} = 1\}.$$

The estimated mean U_{ij} (over the component membership of y_j) is give as

$$\hat{u}_j = \sum_{i=1}^{g} \tau_i(y_j; \hat{\Psi}) \hat{u}_{ij},$$

where $\tau_i(y_j; \hat{\Psi})$ estimated posterior probability of y_j belonging to the ith component.

An alternative estimate of u_j, the posterior expectation of the factor corresponding to the *j*th observation y_j, is defined by replacing $\tau_i(y_j;\, \hat{\Psi})$ by $\hat{z}_{ij}$, where $\hat{z}_{ij} = 1$, if $\hat{\tau}_i(y_j; \hat{\Psi})$ >= $\hat{\tau}_h(y_j; \hat{\Psi})(h = 1, \ldots, g; h \neq i)$, else $\hat{z}_{ij} = 0$.

$$\hat{u}_j^C = \sum_{i=1}^{g} \hat{z}_{ij} \hat{u}_{ij}.$$

For MFA, we have

$$\hat{u}_{ij} = \hat{\beta}_i^T (y_j - \hat{\mu}_i),$$

and

$$\hat{u}_j = \sum_{i=1}^{g} \tau_i(y_j; \hat{\Psi}) \hat{\beta}_i^T (y_j - \hat{\mu}_i)$$

for $j = 1, \ldots, n$ where $\hat{\beta}_i = (B_i B_i^T + D_i)^{-1} B_i$.

For MCFA,

$$\hat{u}_{ij} = \hat{\xi}_i + \hat{\gamma}_i^T (y_j - \hat{A}\hat{\xi}_i),$$

$$\hat{u}_j = \sum_{i=1}^{g} \tau_i(y_j; \hat{\Psi}) \{\hat{\xi}_i + \hat{\gamma}_i^T (y_j - \hat{A}\hat{\xi}_i)\},$$

where $\gamma_i = (A\Omega_i A + D)^{-1} A\Omega_i$.

With M*t*FA and MC*t*FA, the distribution of $\hat{u}_{ij}$ and of $\hat{u}_j$ have the same form as those of MFA and MCFA, respectively.

Usage

```
factor_scores(model, Y, ...)
## S3 method for class 'mcfa'
factor_scores(model, Y, tau = NULL, clust= NULL, ...)
## S3 method for class 'mctfa'
factor_scores(model, Y, tau = NULL, clust= NULL, ...)
## S3 method for class 'emmix'
plot(x, ...)
```

Arguments

`model`	An object of class `mfa,mcfa,mtfa` or `mctfa`.
`x`	An object of class `mfa,mcfa,mtfa` or `mctfa`.
`Y`	Data matrix with variables in columns in the same order as used in model estimation.
`tau`	Optional. Posterior probabilities of belonging to the components in the mixture model. If not provided, they will be computed based on the `model` parameters.

clust — Optional. Indicators of belonging to the components. If not provided, will be estimated using tau.

... — Not used.

Details

Factor scores can be used in visualization of the data in the factor space.

Value

Uscores — Estimated conditional expected component scores of the unobservable factors given the data and the component membership ($\hat{u}_{ij}$). Size is $n \times q \times g$, where n is the number of sample, q is the number of factors and g is the number components.

Umean — Means of the estimated conditional expected factors scores over estimated posterior distributions ($\hat{u}_j$). Size $n \times q$.

Uclust — Alternative estimate of Umean where the posterior probabilities for each sample are replaced by component indicator vectors which contain one in the element corresponding to the highest posterior probability while others zero ($\hat{u}_j^C$). Size $n \times q$.

Author(s)

Geoff McLachlan, Suren Rathnayake, Jungsun Baek

References

McLachlan GJ, Baek J, and Rathnayake SI (2011). Mixtures of factor analyzers for the analysis of high-dimensional data. In *Mixture Estimation and Applications*, KL Mengersen, CP Robert, and DM Titterington (Eds). Hoboken, New Jersey: Wiley, pp. 171–191.

McLachlan GJ, and Peel D (2000). *Finite Mixture Models*. New York: Wiley.

Examples

```
# Fit a MCFA model to a subset
set.seed(1)
samp_size <- dim(iris)[1]
sel_subset <- sample(1 : samp_size, 50)
model <- mcfa(iris[sel_subset, -5], g = 3, q = 2,
                          nkmeans = 1, nrandom = 0, itmax = 100)

# plot the data points in the factor space
plot(model)

# Allocating new samples to the clusters
Y <- iris[-c(sel_subset), -5]
Y <- as.matrix(Y)
clust <- predict(model, Y)

fa_scores <- factor_scores(model, Y)
```

```
# Visualizing new data in factor space
plot_factors(fa_scores, type = "Umean", clust = clust)
```

---

gmf *General Matrix Factorization*

Description

Performs a matrix factorization on the given data set. The factorization is done using a stochastic gradient decent method.

Usage

```
gmf(Y, q, maxit = 1000, lambda = 0.01, cor_rate = 0.9)
```

Arguments

Y	data matrix containing all numerical values.
maxit	maximum number of iterations.
q	number of factors.
lambda	initial learning rate.
cor_rate	correction rate.

Details

Unsupervised matrix factorization of a $n \times p$ data matrix Y can be expressed as,

$$Y^\top \approx AB^\top,$$

where A is a $p \times q$ matrix and B is $n \times q$ matrix. With this matrix factorization method, one replaces the ith row in matrix Y by the ith row in matrix B. The matrices A and B are chosen to minimize an objective function $f(Y, A, B)$ with under constraints specific to the matrix factorization method.

It is imperative that columns of the data matrix be on the same scale. Otherwise, it may not be possible to obtain a factorization of the data using this approach.

Value

A list containing,

A	A numeric matrix of size $p \times q$
B	A numeric matrix of size $n \times q$ matrix

References

Nikulin V, Huang T-H, Ng SK, Rathnayake SI, & McLachlan GJ (2011). A very fast algorithm for matrix factorization. *Statistics & Probability Letters* **81**, 773–782.

Examples

```
lst <- gmf(iris[, -5], q = 2, maxit = 100)
```

mcfa *Mixture of Common Factor Analyzers*

Description

Functions for fitting mixtures of common factor analyzers (MCFA) models. MCFA models are mixture of factor analyzers (belong to the class of multivariate finite mixture models) with a common component matrix for the factor loadings before the transformation of the latent factors to be white noise. It is designed specifically for the task of displaying the observed data points in a lower (q-dimensional) space, where q is the number of factors adopted in the factor-analytic representation of the observed vector.

The `mcfa` function fits mixtures common factor analyzers where the components distributions belong to the family of multivariate normal distributions. The `mctfa` function fits mixtures of common t-factor analyzers where the component distributions corresponds to multivariate t distributions. Maximum likelihood estimates of the model parameters are obtained using the Expectation–Maximization algorithm.

Usage

```
mcfa(Y, g, q, itmax = 500, nkmeans = 5, nrandom = 20,
  tol = 1.e-5, init_clust = NULL, init_para = NULL,
  init_method = NULL, conv_measure = 'diff',
  warn_messages = TRUE, ...)
mctfa(Y, g, q, itmax = 500, nkmeans = 5, nrandom = 20,
  tol = 1.e-5, df_init = rep(30, g), df_update = TRUE,
  init_clust = NULL, init_para = NULL, init_method = NULL,
  conv_measure = 'diff', warn_messages = TRUE, ...)
```

Arguments

`Y`	A matrix or a data frame of which rows correspond to observations and columns to variables.
`g`	Number of components.
`q`	Number of factors.
`itmax`	Maximum number of EM iterations.
`nkmeans`	The number of times the k-means algorithm to be used in partition the data into g groups. These groupings are then used in initializing the parameters for the EM algorithm.
`nrandom`	The number of random g-group partitions for the data to be used initializing the EM algorithm.
`tol`	The EM algorithm terminates if the measure of convergence falls below this value.
`init_clust`	A vector or matrix consisting of partition of samples to be used in the EM algorithm. For matrix of partitions, columns must corresponds individual partitions of the data. Optional.

init_para	A list containing model parameters to be used as initial parameter estimates for the EM algorithm. Optional.
init_method	To determine how the initial parameter values are computed. See Details.
conv_measure	The default 'diff' stops the EM iterations if $\|l^{(k+1)} - l^{(k)}\|$ < tol where $l^{(k)}$ is the log-likelihood at the kth EM iteration. If 'ratio', then the convergence of the EM steps is measured using the $\|(l^{(k+1)} - l^{(k)})/l^{(k+1)}\|$.
df_init	Initial values of the degree of freedom parameters for mctfa.
df_update	If df_update = TRUE (default), then the degree of freedom parameters values will be updated during the EM iterations. Otherwise, if df_update = FALSE, they will be fixed at the initial values specified in df_init.
warn_messages	With warn_messages = TRUE (default), the output would include some description of the reasons where, if any, the model fitting function failed to provide a fit for a given set of initial parameter values.
...	Not used.

Details

With init_method = NULL, the default, model parameters are initialized using all available methods. With the init_method = "rand-A", the initialization of the parameters is done using the procedure in Baek et al. (2010) where initial values for elements of A are drawn from the $N(0, 1)$ distribution. This method is appropriate when the columns of the data are on the same scale. The init_method = "eigen-A" takes the first q eigenvectors of Y as the initial value for the loading matrix A. If init_method = "gmf" then the data are factorized using gmf with q factors and the resulting loading matrix is used as the initial value for A.

If specified, the optional argument init_para must be a list or an object of class mcfa or mctfa. When fitting an mcfa model, only the model parameters q, g, pivec, A, xi, omega, and D are extracted from init_para, while one extra parameter nu is extracted when fitting mctfa. Everything else in init_para will be discarded.

Value

Object of class c("emmix","mcfa") or c("emmix","mctfa") containing the fitted model parameters is returned. Details of the components are as follows:

g	Number of mixture components.
q	Number of factors.
pivec	Mixing proportions of the components.
A	Loading matrix. Size $p \times q$.
xi	Matrix containing factor means for components in columns. Size $q \times g$.
omega	Array containing factor covariance matrices for components. Size $q \times q \times g$.
D	Error covariance matrix. Size $p \times p$.
Uscores	Estimated conditional expected component scores of the unobservable factors given the data and the component membership. Size $n \times q \times g$.
Umean	Means of the estimated conditional expected factors scores over estimated posterior distributions. Size $n \times q$.

`Uclust`	Alternative estimate of `Umean` where the posterior probabilities for each sample are replaced by component indicator vectors which contain one in the element corresponding to the highest posterior probability while others zero. Size $n \times q$.
`clust`	Cluster labels.
`tau`	Posterior probabilities.
`logL`	Log-likelihood at the convergence.
`BIC`	Bayesian information criterion.
`warn_msg`	Description of error messages, if any.

Author(s)

Suren Rathnayake, Jangsun Baek, Geoff McLachlan

References

Baek J, McLachlan GJ, and Flack LK (2010). Mixtures of factor analyzers with common factor loadings: applications to the clustering and visualisation of high-dimensional data. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **32**, 2089–2097.

Baek J, and McLachlan GJ (2011). Mixtures of common *t*-factor analyzers for clustering highdimensional microarray data. *Bioinformatics* **27**, 1269–1276.

McLachlan GJ, Baek J, and Rathnayake SI (2011). Mixtures of factor analyzers for the analysis of high-dimensional data. In *Mixture Estimation and Applications*, KL Mengersen, CP Robert, and DM Titterington (Eds). Hoboken, New Jersey: Wiley, pp. 171–191.

See Also

`mfa`, `plot_factors`

Examples

```
mcfa_fit <- mcfa(iris[, -5], g = 3, q = 3, itmax = 25,
                  nkmeans = 5, nrandom = 5, tol = 1.e-5)

plot(mcfa_fit)

mctfa_fit <- mcfa(iris[, -5], g = 3, q = 3, itmax = 500,
                  nkmeans = 5, nrandom = 5, tol = 1.e-5, df_update = TRUE)
```

mfa *Mixtures of Factor Analyzers*

Description

Functions for fitting mixtures of factor analyzers (MFA) and mixtures of *t*-factor analyzers (M*t*FA) to data. Maximum Likelihood estimates of the model parameters are obtained using the Alternating Expectation Conditional Maximization (AECM) algorithm.

In the case of MFA, component distributions belong to the family of multivariate normal distributions, while with M*t*FA the component distributions correspond to multivariate *t* distributions.

Usage

```
mfa(Y, g, q, itmax = 500, nkmeans = 20, nrandom = 20,
  tol = 1.e-5, sigma_type = 'common', D_type = 'common', init_clust = NULL,
  init_para = NULL, conv_measure = 'diff', warn_messages = TRUE, ...)
mtfa(Y, g, q, itmax = 500, nkmeans = 20, nrandom = 20,
  tol = 1.e-5, df_init = rep(30, g), df_update = TRUE,
  sigma_type = 'common', D_type = 'common', init_clust = NULL,
  init_para = NULL, conv_measure = 'diff', warn_messages = TRUE, ...)
```

Arguments

`Y`	A matrix or a data frame of which rows correspond to observations and columns to variables.
`g`	Number of components.
`q`	Number of factors.
`itmax`	Maximum number of EM iterations.
`nkmeans`	The number of times the k-means algorithm to be used in partition the data into `g` groups. These groupings are then used in initializing the parameters for the EM algorithm.
`nrandom`	The number of random g-group partitions for the data to be used initializing the EM algorithm.
`tol`	The EM algorithm terminates if the measure of convergence falls below this value.
`sigma_type`	To specify whether the covariance matrices (for `mfa`) or the scale matrices (for `mtfa`) of the components are constrained to be the same (default, `sigma_type = "common"`) or not (`sigma_type = "unique"`).
`D_type`	To specify whether the diagonal error covariance matrix is common to all the components or not. If `sigma_type = "unique"`, then `D_type` could either be `"common"` (the default) to each component, or `"unique"`. If the `sigma_type = "common"`, then `D_type` must also be `"common"`.
`init_clust`	A vector or matrix consisting of partition of samples to be used in the EM algorithm. For matrix of partitions, columns must corresponds individual partitions of the data. Optional.

`init_para` A list containing model parameters to be used as initial parameter estimates for the EM algorithm. Optional.

`conv_measure` The default `'diff'` stops the EM iterations if $|l^{(k+1)}$ - $l^{(k)}| <$ `tol` where $l^{(k)}$ is the log-likelihood at the kth EM iteration. If `'ratio'`, then the convergence of the EM steps is measured using the $|(l^{(k+1)}$ - $l^{(k)})/l^{(k+1)}|$.

`df_init` Initial values of the degree of freedom parameters for `mtfa`.

`df_update` If `df_update = TRUE` (default), then the degree of freedom parameters values will be updated during the EM iterations. Otherwise, if `df_update = FALSE`, they will be fixed at the initial values specified in `df_init`.

`warn_messages` With `warn_messages = TRUE` (default), the output would include some description of the reasons where, if any, the model fitting function failed to provide a fit for a given set of initial parameter values.

`...` Not used.

Details

Cluster a given data set using mixtures of factor analyzers or approach or using mixtures of t-factor analyzers.

Value

Object of class `c("emmix","mfa")` or `c("emmix","mtfa")` containing the fitted model parameters is returned. Details of the components are as fellows:

`g` Number of mixture components.

`q` Number of factors.

`pivec` Mixing proportions of the components.

`mu` Matrix containing estimates of component means (in columns) of mixture component. Size $p \times g$.

`B` Array containing component dependent loading matrices. Size $p \times q \times g$.

`D` Estimates of error covariance matrices. If `D_type = "common"` was used then `D` is $p \times p$ matrix common to all components, if `D_type = "unique"`, then `D` is a size $p \times p \times g$ array.

`v` Degrees of freedom for each component.

`logL` Log-likelihood at the convergence.

`BIC` Bayesian information criterion.

`tau` Matrix of posterior probabilities for the data used based on the fitted values. Matrix of size `n` by `g`.

`clust` Vector of integers 1 to g indicating cluster allocations of the observations.

`Uscores` Estimated conditional expected component scores of the unobservable factors given the data and the component membership. Size is Size $n \times q \times g$.

`Umean` Means of the estimated conditional expected factors scores over estimated posterior distributions. Size $n \times q$.

`Uclust`	Alternative estimate of `Umean` where the posterior probabilities for each sample are replaced by component indicator vectors which contain one in the element corresponding to the highest posterior probability while others zero. Size $n \times q$.
`ERRMSG`	Description of messages, if any.
`D_type`	Whether common or unique error covariance is used, as specified in model fitting.
`df_update`	Whether the degree of freedom parameter (`v`) was fixed or estimated (only for `mtfa`).

Author(s)

Suren Rathnayake, Geoffrey McLachlan

References

Ghahramani Z, and Hinton GE (1997). The EM algorithm for mixture of factor analyzers. *Technical Report, CRG-TR-96-1*, University of Toronto, Toronto.

McLachlan GJ, Bean RW, Ben-Tovim Jones L (2007). Extension of the mixture of factor analyzers model to incorporate the multivariate *t* distribution. *Computational Statistics & Data Analysis*, **51**, 5327–5338.

McLachlan GJ, Baek J, and Rathnayake SI (2011). Mixtures of factor analyzers for the analysis of high-dimensional data. In *Mixture Estimation and Applications*, KL Mengersen, CP Robert, and DM Titterington (Eds). Hoboken, New Jersey: Wiley, pp. 171–191.

McLachlan GJ, Peel D, and Bean RW (2003). Modelling high-dimensional data by mixtures of factor analyzers. *Computational Statistics & Data Analysis* **41**, 379–388.

See Also

`mcfa`

Examples

```
model <- mfa(iris[, -5], g=3, q=2, itmax=200, nkmeans=1, nrandom=5)
summary(model)

model <- mtfa(iris[, -5], g=3, q=2, itmax=200, nkmeans=1, nrandom=5)
```

`minmis` *Minimum Number of Misallocations*

Description

Given two vectors each corresponding to a set of categories, this function finds the minimum number of misallocations by rotating the categories.

Usage

```
minmis(cls, hat_cls)
```

Arguments

`cls`	A numeric or character vector of labels.
`hat_cls`	A numeric or character vector of labels same length as `cls`.

Details

Rotates the categories for all possible permutations, and returns the minimum number of misallocations. The number of categories in each set of labels does not need to be the same. It may take several minutes to compute when the number of categories is large.

Value

Integer specifying the minimum number of misallocations.

See Also

`ari`

Examples

```
set.seed(1984)
Y <- scale(iris[, -5])
model <- mcfa(Y, g = 3, q = 3, nkmeans = 1, nrandom = 0, itmax = 200)
ari(model$clust, iris[, 5])
minmis(model$clust, iris[, 5])
```

`plot_factors` *Plot Function for Factor Scores*

Description

Plot functions for factor scores.

Usage

```
plot_factors(scores, type = "Umean",
    clust=if (exists('clust', where = scores)) scores$clust else NULL,
    limx = NULL, limy = NULL)
```

Arguments

scores	A list containing factor scores specified by Umean, Uclust or Uscores, or a model of class mcfa,mctfa,mfa, or mtfa.
type	What type of factor scores are to be plotted. See Details.
clust	Indicators of belonging to components. If available, they will be portrayed in plots. If not provided, looks for clust in scores, and sets to NULL if still not available.
limx	Numeric vector. Values in limx will only be used in setting the x-axis range for 1-D and 2-D plots.
limy	Numeric vector. Values in limy will only be used in setting the y-axis range for 1-D and 2-D plots.

Details

When the factor scores were obtained using mcfa or mctfa, then a visualization of the group structure can be obtained by plotting the factor scores. In the case of mfa and mtfa, the factor scores simply corresponds to white noise.

The type should either be "Uscores", "Uclust" or the default "Umean". See factor_scores for a detailed description of the factor scores.

Author(s)

Geoffrey McLachlan, Suren Rathnayake, Jungsun Baek

References

McLachlan GJ, Baek J, and Rathnayake SI (2011). Mixtures of factor analyzers for the analysis of high-dimensional data. In *Mixture Estimation and Applications*, KL Mengersen, CP Robert, and DM Titterington (Eds). Hoboken, New Jersey: Wiley, pp. 171–191.

McLachlan GJ, and Peel D (2000). *Finite Mixture Models*. New York: Wiley.

Examples

```
# Visualizing data used in model estimation
set.seed(1)
inds <- dim(iris)[1]
indSample <- sample(1 : inds, 50)
model <- mcfa (iris[indSample, -5], g = 3, q = 2,
                nkmeans = 1, nrandom = 0, itmax = 150)
minmis(model$clust, iris[indSample, 5])

#same as plot_factors(model, tyep = "Umean", clust = model$clust)
plot(model)

#can provide alternative groupings of samples via plot_factors
plot_factors(model, clust = iris[indSample, 5])

#same as plot_factors(model, tyep = "Uclust")
```

```
plot(model, type = "Uclust")

Y <- iris[-c(indSample), -5]
Y <- as.matrix(Y)
clust <- predict(model, Y)
minmis(clust, iris[-c(indSample), 5])

fac_scores <- factor_scores(model, Y)
plot_factors(fac_scores, type = "Umean", clust = clust)
plot_factors(fac_scores, type = "Umean", clust = iris[-c(indSample), 5])
```

`predict.emmix` *Extend Clustering to New Observations*

Description

Given a fitted model of class `'emmix'` (or of class `'mfa'`, `'mcfa'`, `'mtfa'` and `'mctfa'`), the `predict` function predict clusters for observations.

Usage

```
## S3 method for class 'emmix'
predict(object, Y, ...)
```

Arguments

`object`	An object of class `'emmix'`.
`Y`	A data matrix with variable in the same column locations as the data used in fitting the model `object`.
`...`	Not used.

Details

A vector integers of length equal to number of observations (rows) in the data. The integers range from 1 to g where g in the number of components in the model.

The variables in `Y` of the `predict` function should be in the order as those used in obtaining the fitted model `object`.

Examples

```
set.seed(42)
test <- sample(1 : nrow(iris), 100)
model <- mfa(iris[test, -5], g=3, q=3, itmax=500, nkmeans=3, nrandom=5)
pred_clust <- predict(model, iris[-test, -5])
minmis(pred_clust, iris[-test, 5])
```

print.emmix — *Print Method for Class 'emmix'*

Description

Prints a formatted model parameters of `EMMIXmfa` objects.

Usage

```
## S3 method for class 'emmix'
print(x, ...)
## S3 method for class 'emmix'
summary(object, ...)
```

Arguments

`x, object`	An object of class `'emmix'`.
`...`	Not used.

Details

Prints the formatted model parameter values to the screen.

Examples

```
set.seed(1984)
Y <- scale(iris[, -5])
model <- mcfa(Y, g = 3, q = 3, nkmeans = 1, nrandom = 0, itmax = 100)
#
print(model)
summary(model)
```

rmix — *Random Deviates from EMMIX Models*

Description

Random number generator for `emmix` models.

Usage

```
rmix(n, model, ...)
```

Arguments

model	An object of class 'emmix' containing a mode of mfa,mcfa,mtfa, or mctfa.
n	Number of sample to generate.
...	Not used.

Details

This function uses rmvnorm and rmvt functions from the **mvtnorm** package to generate samples from the mixture components.

Algorithm works by first drawing a component based on the mixture proprotion in the model, and then drawing a sample from the component distribution.

Value

A numeric matrix with samples drawn in rows.

Examples

```
set.seed(1)
model <- mcfa(iris[, -5], g=3, q=2, nkmeans=1, nrandom=1, itmax = 25)
dat <- rmix(n = 10, model = model)
```

Index