Package: mixture (via r-universe)

August 27, 2024

Type Package

Title Mixture Models for Clustering and Classification

Version 2.1.1

Date 2024-01-29

Maintainer Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

Description An implementation of 14 parsimonious mixture models for model-based clustering or model-based classification. Gaussian, Student's t, generalized hyperbolic, variance-gamma or skew-t mixtures are available. All approaches work with missing data. Celeux and Govaert (1995) <doi:10.1016/0031-3203(94)00125-6>, Browne and McNicholas (2014) <doi:10.1007/s11634-013-0139-1>, Browne and McNicholas (2015) <doi:10.1002/cjs.11246>.

Repository CRAN

LazyLoad yes

NeedsCompilation yes

License GPL (>= 2)

Imports Rcpp (>= 1.0.2), methods

LinkingTo Rcpp, RcppArmadillo, BH, RcppGSL

Depends R (>= 3.5.0), lattice (>= 0.20)

SystemRequirements GNU GSL

Author Nik Pocuca [aut] (<https://orcid.org/0000-0001-8365-0751>), Ryan P. Browne [aut] (<https://orcid.org/0000-0003-4543-0218>), Paul D. McNicholas [aut, cre] (<https://orcid.org/0000-0002-2482-523X>)

Date/Publication 2024-01-30 00:20:07 UTC

Contents

`ARI` *Adjusted Rand Index*

Description

Calculates an adjusted for chance Rand index.

Usage

```
ARI(x,y)
```

Arguments

`x`	predictor class memberships
`y`	true class memberships

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

Examples

```
x <- sample(1:10, size = 100, replace = TRUE)
y <- sample(1:10, size = 100, replace = TRUE)
ARI(x,y)
```

`e_step` *Expectation Step*

Description

Calculates the expectation of class memberships, and imputes if missing values for a given dataset.

Usage

```
e_step(data, model_obj, start=0, nu = 1.0)
```

Arguments

`data`	A matrix or data frame such that rows correspond to observations and columns correspond to variables. Note that this function currently only works with multivariate data p > 1.
`start`	Start values in this context are only used for imputation. Non-missing values have their expectation of class memberships calculated directly. If `0` then the random soft function is used for initialization. If `1` then the random hard function is used for initialization. If `2` then the kmeans function is used for initialization. If `is.matrix` then matrix is used as an initialization matrix as along as it has non-negative elements. Note: only models with the same number of columns of this matrix will be fit.
`model_obj`	A gpcm_best, vgpcm_best, stpcm_best, ghpcm_best, and salpcm_best object class.
`nu`	deterministic annealing for the class membership E-step.

Details

This will only work on a dataset with the same dimension as estimated in the model. `e_step` will also work for missing values, provided that there is at least one non-missing entry.

Value

Returns a list with the following components:

`X`	A matrix of the original dataset plus imputed values if applicable.
`origX`	A matrix of the original dataset including missing values.
`map`	A vector of integers indicating the maximum *a posteriori* classifications for the best model.
`z`	A matrix giving the raw values upon which `map` is based.
`row_tags`	If there were NAs in the original dataset, a vector of indices referencing the row of the imputed vectors is given.

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Zhou, H. and Lange, K. (2010). On the bumpy road to the dominant mode. *Scandinavian Journal of Statistics* **37**, 612-631.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Examples

```
## Not run:
# load dataset and perform model search.

data(x2)
data_in <- matrix(x2,ncol = 2)
mm <- mixture::gpcm(data = data_in,G = 1:7,
           start = 0,
           veo = FALSE,pprogress=FALSE)

# get best model
best = get_best_model(mm)
best

# lets try imputing some missing data.
x2NA <- x2
x2NA[5,1] <- NA
x2NA[140,2] <- NA
x2NA[99,1] <- NA

# calculate expectation
expect <- e_step(data=x2NA,start = 0,nu = 1.0,model_obj = best)

# plot imputed entries and compare with original
plot(x2,col = "grey")
points(expect$X[expect$row_tags+1,],col = "blue", pch = 20,cex = 2) # blue are imputed values.
points(x2[expect$row_tags+1,], col = "red" , pch = 20,cex = 2) # red are original values.
legend(-2,2,legend = c("imputed","original"),col = c("blue","red"),pch = 20)

## End(Not run)
```

get_best_model *Best Model Extractor*

Description

Carries out model-based clustering or classification using some or all of the 14 parsimonious Gaussian clustering models (GPCM).

Usage

```
get_best_model(gpcm_model)
```

Arguments

gpcm_model	An input of class gpcm.

Details

Extracts the best model based on BIC.

Value

An object of class gpcm_best is a list with components:

model_type	A string containg summarized information about the type of model estimated (Covariance structure and number of groups).
model_obj	An internal list containing all parameters returned from the C++ call.
BIC	Bayesian Index Criterion (positive scale, bigger is better).
loglik	Log liklihood from the estimated model.
nparam	Number of a parameters in the mode.
startobject	The type of object inputted into start.
G	An integer representing the number of groups.
cov_type	A string representing the type of covariance matrix (see 14 models).
status	Convergence status of EM algorithm according to Aitken's Acceleration
map	A vector of integers indicating the maximum *a posteriori* classifications for the best model.
row_tags	If there were NAs in the original dataset, a vector of indices referencing the row of the imputed vectors is given.

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Zhou, H. and Lange, K. (2010). On the bumpy road to the dominant mode. *Scandinavian Journal of Statistics* **37**, 612-631.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Examples

```
## Not run:

# load dataset and perform model search.
data(x2)
data_in <- matrix(x2,ncol = 2)
mm <- mixture::gpcm(data = data_in,G = 1:7,
           start = 0,
           veo = FALSE,pprogress=FALSE)

# get best model
best = get_best_model(mm)
best

## End(Not run)
```

---

ghpcm *Generalized Hyperbolic Parsimonious Clustering Models*

Description

Carries out model-based clustering or classification using some or all of the 14 parsimonious Generalized Hyperbolic clustering models (GHPCM).

Usage

```
ghpcm(data=NULL, G=1:3, mnames=NULL,
  start=2, label=NULL,
  veo=FALSE, da=c(1.0),
  nmax=1000, atol=1e-8, mtol=1e-8, mmax=10, burn=5,
  pprogress=FALSE, pwarning=FALSE, stochastic = FALSE, seed=123)
```

Arguments

`data`	A matrix or data frame such that rows correspond to observations and columns correspond to variables. Note that this function currently only works with multivariate data $p > 1$.
`G`	A sequence of integers giving the number of components to be used.

mnames	The models (i.e., covariance structures) to be used. If NULL then all 14 are fitted.
start	If 0 then the random soft function is used for initialization. If 1 then the random hard function is used for initialization. If 2 then the kmeans function is used for initialization. If >2 then multiple random soft starts are used for initialization. If is.matrix then matrix is used as an initialization matrix as along as it has non-negative elements. Note: only models with the same number of columns of this matrix will be fit.
label	If NULL then the data has no known groups. If is.integer then some of the observations have known groups. If label[i]=k then observation belongs to group k. If label[i]=0 then observation has no known group. See Examples.
veo	Stands for "Variables exceed observations". If TRUE then if the number variables in the model exceeds the number of observations the model is still fitted.
da	Stands for Determinstic Annealing. A vector of doubles.
nmax	The maximum number of iterations each EM algorithm is allowed to use.
atol	A number specifying the epsilon value for the convergence criteria used in the EM algorithms. For each algorithm, the criterion is based on the difference between the log-likelihood at an iteration and an asymptotic estimate of the log-likelihood at that iteration. This asymptotic estimate is based on the Aitken acceleration and details are given in the References.
mtol	A number specifying the epsilon value for the convergence criteria used in the M-step in the GEM algorithms.
mmax	The maximum number of iterations each M-step is allowed in the GEM algorithms.
burn	The burn in period for imputing data. (Missing observations are removed and a model is estimated seperately before placing an imputation step within the EM.)
pprogress	If TRUE print the progress of the function.
pwarning	If TRUE print the warnings.
stochastic	If TRUE , it will run stochastic E step variant.
seed	The seed for the run, default is 123

Details

The data x are either clustered or classified using Generalized Hyperbolic mixture models with some or all of the 14 parsimonious covariance structures described in Celeux & Govaert (1995). The algorithms given by Celeux & Govaert (1995) is used for 12 of the 14 models; the "EVE" and "VVE" models use the algorithms given in Browne & McNicholas (2014). Starting values are very important to the successful operation of these algorithms and so care must be taken in the interpretation of results.

Value

An object of class ghpcm is a list with components:

map	A vector of integers indicating the maximum *a posteriori* classifications for the best model.

model_objs	A list of all estimated models with parameters returned from the C++ call.
best_model	A class of vgpcm_best containing; the number of groups for the best model, the covariance structure, and Bayesian Information Criterion (BIC) value.
loglik	The log-likelihood values from fitting the best model.
z	A matrix giving the raw values upon which map is based.
BIC	A G by mnames by 3 dimensional array with values pertaining to BIC calculations. (legacy)
startobject	The type of object inputted into start.
gpar	A list object for each cluster pertaining to parameters. (legacy)
row_tags	If there were NAs in the original dataset, a vector of indices referencing the row of the imputed vectors is given.

Best Model: An object of class ghpcm_best is a list with components:

model_type	A string containg summarized information about the type of model estimated (Covariance structure and number of groups).
model_obj	An internal list containing all parameters returned from the C++ call.
BIC	Bayesian Index Criterion (positive scale, bigger is better).
loglik	Log liklihood from the estimated model.
nparam	Number of a parameters in the mode.
startobject	The type of object inputted into start.
G	An integer representing the number of groups.
cov_type	A string representing the type of covariance matrix (see 14 models).
status	Convergence status of EM algorithm according to Aitken's Acceleration
map	A vector of integers indicating the maximum *a posteriori* classifications for the best model.
row_tags	If there were NAs in the original dataset, a vector of indices referencing the row of the imputed vectors is given.

Internal Objects: All classes contain an internal list called model_obj or model_objs with the following components:

zigs	a posteori matrix
G	An integer representing the number of groups.
sigs	A vector of covariance matrices for each group
mus	A vector of location vectors for each group
alphas	A vector containg skewness vectors for each group
gammas	A vector containing estimated gamma parameters for each group

Note

Dedicated print, plot and summary functions are available for objects of class ghpcm.

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

McNicholas, P.D. (2016), *Mixture Model-Based Classification*. Boca Raton: Chapman & Hall/CRC Press

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Browne, R.P. and McNicholas, P.D. (2015), 'A mixture of generalized hyperbolic distributions', Canadian Journal of Statistics 43(2), 176-198.

Zhou, H. and Lange, K. (2010). On the bumpy road to the dominant mode. *Scandinavian Journal of Statistics* **37**, 612-631.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Examples

```
 ## Not run:

data("sx2")


### use random soft initializations.
ax6 = ghpcm(sx2, G=1:3,start= 0)
summary(ax6)
ax6

### plot results
plot(sx2,col = ax6$map + 1)

### use deterministic annealing for starting values
axDA = ghpcm(sx2, G=1:3, start=0,da=c(0.3,0.5,0.8,1.0))
summary(axDA)
axDA


## End(Not run)
```

---

gpcm *Gaussian Parsimonious Clustering Models*

Description

Carries out model-based clustering or classification using some or all of the 14 parsimonious Gaussian clustering models (GPCM).

Usage

```
gpcm(data=NULL, G=1:3, mnames=NULL,
  start=2, label=NULL,
  veo=FALSE, da=c(1.0),
  nmax=1000, atol=1e-8, mtol=1e-8, mmax=10, burn=5,
  pprogress=FALSE, pwarning=TRUE, stochastic = FALSE, seed=123)
```

Arguments

`data`	A matrix or data frame such that rows correspond to observations and columns correspond to variables. Note that this function currently only works with multivariate data p > 1.
`G`	A sequence of integers giving the number of components to be used.
`mnames`	The models (i.e., covariance structures) to be used. If `NULL` then all 14 are fitted.
`start`	If `0` then the random soft function is used for initialization. If `1` then the random hard function is used for initialization. If `2` then the kmeans function is used for initialization. If `>2` then multiple random soft starts are used for initialization. If `is.matrix` then matrix is used as an initialization matrix as along as it has non-negative elements. Note: only models with the same number of columns of this matrix will be fit.
`label`	If `NULL` then the data has no known groups. If `is.integer` then some of the observations have known groups. If `label[i]=k` then observation belongs to group `k`. If `label[i]=0` then observation has no known group. See Examples.
`veo`	Stands for "Variables exceed observations". If `TRUE` then if the number variables in the model exceeds the number of observations the model is still fitted.
`da`	Stands for Determinstic Annealing. A vector of doubles.
`nmax`	The maximum number of iterations each EM algorithm is allowed to use.
`atol`	A number specifying the epsilon value for the convergence criteria used in the EM algorithms. For each algorithm, the criterion is based on the difference between the log-likelihood at an iteration and an asymptotic estimate of the log-likelihood at that iteration. This asymptotic estimate is based on the Aitken acceleration and details are given in the References.
`mtol`	A number specifying the epsilon value for the convergence criteria used in the M-step in the GEM algorithms.
`mmax`	The maximum number of iterations each M-step is allowed in the GEM algorithms.
`burn`	The burn in period for imputing data. (Missing observations are removed and a model is estimated seperately before placing an imputation step within the EM.)
`pprogress`	If `TRUE` print the progress of the function.
`pwarning`	If `TRUE` print the warnings.
`stochastic`	If `TRUE` , it will run stochastic E step variant.
`seed`	The seed for the run, default is 123

Details

The data x are either clustered or classified using Gaussian mixture models with some or all of the 14 parsimonious covariance structures described in Celeux & Govaert (1995). The algorithms given by Celeux & Govaert (1995) is used for 12 of the 14 models; the "EVE" and "VVE" models use the algorithms given in Browne & McNicholas (2014). Starting values are very important to the successful operation of these algorithms and so care must be taken in the interpretation of results.

Value

An object of class gpcm is a list with components:

- map — A vector of integers indicating the maximum *a posteriori* classifications for the best model.
- model_objs — A list of all estimated models with parameters returned from the C++ call.
- best_model — A class of gpcm_best containing; the number of groups for the best model, the covariance structure, and Bayesian Information Criterion (BIC) value.
- loglik — The log-likelihood values from fitting the best model.
- z — A matrix giving the raw values upon which map is based.
- BIC — A G by mnames by 3 dimensional array with values pertaining to BIC calculations. (legacy)
- gpar — A list object for each cluster pertaining to parameters. (legacy)
- startobject — The type of object inputted into start.
- row_tags — If there were NAs in the original dataset, a vector of indices referencing the row of the imputed vectors is given.

Best Model: An object of class gpcm_best is a list with components:

- model_type — A string containg summarized information about the type of model estimated (Covariance structure and number of groups).
- model_obj — An internal list containing all parameters returned from the C++ call.
- BIC — Bayesian Index Criterion (positive scale, bigger is better).
- loglik — Log liklihood from the estimated model.
- nparam — Number of a parameters in the mode.
- startobject — The type of object inputted into start.
- G — An integer representing the number of groups.
- cov_type — A string representing the type of covariance matrix (see 14 models).
- status — Convergence status of EM algorithm according to Aitken's Acceleration
- labs — A vector of integers indicating the maximum *a posteriori* classifications for the best model.
- row_tags — If there were NAs in the original dataset, a vector of indices referencing the row of the imputed vectors is given.

Internal Objects: All classes contain an internal list called model_obj or model_objs with the following components:

zigs	a posteori matrix
G	An integer representing the number of groups.
sigs	A vector of covariance matrices for each group
mus	A vector of mean vectors for each group

Note

Dedicated `print`, `plot` and `summary` functions are available for objects of class `gpcm`.

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

McNicholas, P.D. (2016), *Mixture Model-Based Classification*. Boca Raton: Chapman & Hall/CRC Press

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Examples

```
 ## Not run:

data("x2")
### use kmeans to find starting values
ax0 = gpcm(x2, G=1:5, mnames=c("VVV", "EVE"),start=2, pprogress=TRUE, atol=1e-2)
summary(ax0)
ax0

### use random soft initializations.
ax6 = gpcm(x2, G=1:5, mnames=c("VVV", "EVE"),start= 0)
summary(ax6)
ax6

### use deterministic annealing for starting values
axDA = gpcm(x2, G=1:5, mnames=c("VVV", "EVE"), start=0,da=c(0.3,0.5,0.8,1.0))
summary(axDA)
axDA

### estimate all 14 covariance structures
ax = gpcm(x2, G=1:5, mnames=NULL, start=0)
summary(ax)
ax

### model based classification
x2.label = numeric(nrow(x2))
```

```
x2.label[c(10,50, 110, 150, 210, 250)] = c(1,1,2,2,3,3)
axl = gpcm(x2, G=3, mnames=c("VVV", "EVE"), label=x2.label)
summary(axl)

plot(x2, col = axl$map + 1)

## End(Not run)
```

---

main_loop *GPCM Internal C++ Call*

Description

This function is the internal C++ function call within the gpcm function. This is a raw C++ function call, meaning it has no checks for proper inputs so it may fail to run without giving proper errors. Please ensure all arguements are valid. main_loop is useful for writing parallizations of the gpcm function. All arguement descriptions are given in terms of their corresponding C++ types.

Usage

```
main_loop(X, G, model_id,
        model_type, in_zigs,
        in_nmax, in_l_tol, in_m_iter_max,
        in_m_tol, anneals, t_burn = 5L)
```

Arguments

X	A matrix or data frame such that rows correspond to observations and columns correspond to variables. Note that this function currently only works with multivariate data p > 1.
G	A single positive integer value representing number of groups.
model_id	An integer representing the model_id, is useful for keeping track within paralizations. Not to be confused with model_type.
model_type	The type of covariance model you wish to run. Lexicon is given as follows: "0" = "EII", "1" = "VII", "2" = "EEI" , "3" = "EVI", "4" = "VEI", "5" = "VVI", "6" = "EEE", "7" = "VEE", "8" = "EVE", "9" = "EEV", "10" = "VVE", "11" = "EVV", "12" = "VEV", "13" = "VVV"
in_zigs	A n times G a posteriori matrix resembling the probability of observation i belonging to group G. Rows must sum to one, have the proper dimensions, and be positive.
in_nmax	Positive integer value resembling the maximum amount of iterations for the EM.
in_l_tol	A likelihood tolerance for convergence.
in_m_iter_max	For certain models, where applicable, the number of iterations for the maximization step.
in_m_tol	For certain models, where applicable, the tolerance for the maximization step.

anneals A vector of doubles representing the deterministic annealing settings.

t_burn A positive integer representing the number of burn steps if missing data (NAs) are detected.

Details

Be extremly careful running this function, it is known to crash systems without proper exception handling. Consider using the package `parallel` to estimate all possible models at the same time.

Value

zigs a postereori matrix

G An integer representing the number of groups.

sigs A vector of covariance matrices for each group (note you may have to reshape this)

mus A vector of mean vectors for each group

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Zhou, H. and Lange, K. (2010). On the bumpy road to the dominant mode. *Scandinavian Journal of Statistics* **37**, 612-631.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Examples

```
## Not run:

data("x2")
data_in = as.matrix(x2,ncol = 2)
n_iter = 1000

in_g = 3
n = dim(data_in)[1]
model_string <- "VVE"
in_model_type <- switch(model_string, "EII" = 0,"VII" = 1,
             "EEI" = 2,  "EVI" = 3,  "VEI" = 4,  "VVI" = 5,  "EEE" = 6,
             "VEE" = 7,  "EVE" = 8,  "EEV" = 9,  "VVE" = 10,
             "EVV" = 11,"VEV" = 12,"VVV" = 13)

zigs_in <- z_ig_random_soft(n,in_g)
```

```
m2 = main_loop(X = data_in, # data in
               G = 3, # number of groups
               model_id = 1, # model id for parallelization later
               model_type = in_model_type,
               in_zigs = zigs_in, # initializaiton
               in_nmax = n_iter, # number of iterations
               in_l_tol = 1e-12, # likilihood tolerance
               in_m_iter_max = 20, # maximium iterations for matrices
               in_m_tol = 1e-8,
               anneals=c(0.5,0.7,0.9,1))

plot(data_in,col = MAP(m2$zigs) + 1)

## End(Not run)
```

`main_loop_gh` *GHPCM Internal C++ Call*

Description

This function is the internal C++ function call within the ghpcm function. This is a raw C++ function call, meaning it has no checks for proper inputs so it may fail to run without giving proper errors. Please ensure all arguements are valid. `main_loop_gh` is useful for writing parallizations of the ghpcm function. All arguement descriptions are given in terms of their corresponding C++ types.

Usage

```
main_loop_gh(X, G, model_id,
        model_type, in_zigs,
        in_nmax, in_l_tol, in_m_iter_max,
        in_m_tol, anneals, t_burn = 5L)
```

Arguments

X	A matrix or data frame such that rows correspond to observations and columns correspond to variables. Note that this function currently only works with multivariate data p > 1.
G	A single positive integer value representing number of groups.
model_id	An integer representing the model_id, is useful for keeping track within parallizations. Not to be confused with model_type.
model_type	The type of covariance model you wish to run. Lexicon is given as follows: "0" = "EII", "1" = "VII", "2" = "EEI" , "3" = "EVI", "4" = "VEI", "5" = "VVI", "6" = "EEE", "7" = "VEE", "8" = "EVE", "9" = "EEV", "10" = "VVE", "11" = "EVV", "12" = "VEV", "13" = "VVV"
in_zigs	A n times G a posteriori matrix resembling the probability of observation i belonging to group G. Rows must sum to one, have the proper dimensions, and be positive.

`in_nmax`	Positive integer value resembling the maximum amount of iterations for the EM.
`in_l_tol`	A likelihood tolerance for convergence.
`in_m_iter_max`	For certain models, where applicable, the number of iterations for the maximization step.
`in_m_tol`	For certain models, where applicable, the tolerance for the maximization step.
`anneals`	A vector of doubles representing the deterministic annealing settings.
`t_burn`	A positive integer representing the number of burn steps if missing data (NAs) are detected.

Details

Be extremly careful running this function, it is known to crash systems without proper exception handling. Consider using the package `parallel` to estimate all possible models at the same time. Or run several possible initializations with random seeds.

Value

`zigs`	a postereori matrix
`G`	An integer representing the number of groups.
`sigs`	A vector of covariance matrices for each group (note you may have to reshape this)
`mus`	A vector of locational vectors for each group
`alphas`	A vector of skewness vectors for each group
`omegas`	First set of gamma parameters for each group
`lambdas`	Second set of gamma parameters for each group

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

McNicholas, P.D. (2016), *Mixture Model-Based Classification*. Boca Raton: Chapman & Hall/CRC Press

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Browne, R.P. and McNicholas, P.D. (2015), 'A mixture of generalized hyperbolic distributions', Canadian Journal of Statistics 43(2), 176-198.

Zhou, H. and Lange, K. (2010). On the bumpy road to the dominant mode. *Scandinavian Journal of Statistics* **37**, 612-631.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Examples

```
## Not run:

data("sx2")
data_in = as.matrix(sx2,ncol = 2)
n_iter = 300

in_g = 2
n = dim(data_in)[1]
model_string <- "VVV"
in_model_type <- switch(model_string, "EII" = 0,"VII" = 1,
              "EEI" = 2,  "EVI" = 3,  "VEI" = 4,  "VVI" = 5,  "EEE" = 6,
              "VEE" = 7,  "EVE" = 8,  "EEV" = 9,  "VVE" = 10,
              "EVV" = 11,"VEV" = 12,"VVV" = 13)

zigs_in <- z_ig_random_soft(n,in_g)

m2 = main_loop_gh(X = t(data_in), # data in has to be in column major form
               G = 2, # number of groups
               model_id = 1, # model id for parallelization later
               model_type = in_model_type,
               in_zigs = zigs_in, # initializaiton
               in_nmax = n_iter, # number of iterations
               in_l_tol = 1e-8, # likilihood tolerance
               in_m_iter_max = 20, # maximium iterations for matrices
               in_m_tol = 1e-8,
               anneals=c(0.5,0.7,0.9,1))

plot(sx2,col = MAP(m2$zigs) + 1, cex = 0.5, pch = 20)

## End(Not run)
```

---

main_loop_st *STPCM Internal C++ Call*

Description

This function is the internal C++ function call within the `stpcm` function. This is a raw C++ function call, meaning it has no checks for proper inputs so it may fail to run without giving proper errors. Please ensure all arguements are valid. `main_loop_st` is useful for writing parallizations of the stpcm function. All arguement descriptions are given in terms of their corresponding C++ types.

Usage

```
main_loop_st(X, G, model_id,
        model_type, in_zigs,
        in_nmax, in_l_tol, in_m_iter_max,
        in_m_tol, anneals,
        latent_step="standard",
        t_burn = 5L)
```

Arguments

`X`	A matrix or data frame such that rows correspond to observations and columns correspond to variables. Note that this function currently only works with multivariate data p > 1.
`G`	A single positive integer value representing number of groups.
`model_id`	An integer representing the model_id, is useful for keeping track within parallizations. Not to be confused with model_type.
`model_type`	The type of covariance model you wish to run. Lexicon is given as follows: "0" = "EII", "1" = "VII", "2" = "EEI" , "3" = "EVI", "4" = "VEI", "5" = "VVI", "6" = "EEE", "7" = "VEE", "8" = "EVE", "9" = "EEV", "10" = "VVE", "11" = "EVV", "12" = "VEV", "13" = "VVV"
`in_zigs`	A n times G a posteriori matrix resembling the probability of observation i belonging to group G. Rows must sum to one, have the proper dimensions, and be positive.
`in_nmax`	Positive integer value resembling the maximum amount of iterations for the EM.
`in_l_tol`	A likelihood tolerance for convergence.
`in_m_iter_max`	For certain models, where applicable, the number of iterations for the maximization step.
`in_m_tol`	For certain models, where applicable, the tolerance for the maximization step.
`anneals`	A vector of doubles representing the deterministic annealing settings.
`t_burn`	A positive integer representing the number of burn steps if missing data (NAs) are detected.
`latent_step`	If `"standard"`, it will use the standard E step for latent variable of a Normal Variance Mean Mixture, if `"random"` it will run a random draw from a GIG distribution.

Details

Be extremly careful running this function, it is known to crash systems without proper exception handling. Consider using the package `parallel` to estimate all possible models at the same time. Or run several possible initializations with random seeds.

Value

`zigs`	a postereori matrix
`G`	An integer representing the number of groups.
`sigs`	A vector of covariance matrices for each group (note you may have to reshape this)
`mus`	A vector of locational vectors for each group
`alphas`	A vector of skewness vectors for each group
`vgs`	Gamma parameters for each group

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

McNicholas, P.D. (2016), *Mixture Model-Based Classification*. Boca Raton: Chapman & Hall/CRC Press

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Wei, Y., Tang, Y. and McNicholas, P.D. (2019), 'Mixtures of generalized hyperbolic distributions and mixtures of skew-t distributions for model-based clustering with incomplete data', Computational Statistics and Data Analysis 130, 18-41.

Zhou, H. and Lange, K. (2010). On the bumpy road to the dominant mode. *Scandinavian Journal of Statistics* **37**, 612-631.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Examples

```
## Not run:

data("sx2")
data_in = as.matrix(sx2,ncol = 2)
n_iter = 300

in_g = 2
n = dim(data_in)[1]
model_string <- "VEI"
in_model_type <- switch(model_string, "EII" = 0,"VII" = 1,
              "EEI" = 2,  "EVI" = 3,  "VEI" = 4,  "VVI" = 5,  "EEE" = 6,
              "VEE" = 7,  "EVE" = 8,  "EEV" = 9,  "VVE" = 10,
              "EVV" = 11,"VEV" = 12,"VVV" = 13)

zigs_in <- z_ig_random_soft(n,in_g)

m2 = main_loop_st(X = t(data_in), # data in has to be in column major form
               G = 2, # number of groups
               model_id = 1, # model id for parallelization later
               model_type = in_model_type,
               in_zigs = zigs_in, # initializaiton
               in_nmax = n_iter, # number of iterations
               in_l_tol = 0.5, # likilihood tolerance
               in_m_iter_max = 20, # maximium iterations for matrices
               anneals=c(1),
               in_m_tol = 1e-8)

plot(sx2,col = MAP(m2$zigs) + 1, cex = 0.5, pch = 20)

## End(Not run)
```

main_loop_t *TPCM Internal C++ Call*

Description

This function is the internal C++ function call within the `stpcm` function. This is a raw C++ function call, meaning it has no checks for proper inputs so it may fail to run without giving proper errors. Please ensure all arguements are valid. `main_loop_st` is useful for writing parallizations of the stpcm function. All arguement descriptions are given in terms of their corresponding C++ types.

Usage

```
main_loop_t(X, G, model_id,
        model_type, in_zigs,
        in_nmax, in_l_tol, in_m_iter_max,
        in_m_tol, anneals, t_burn = 5L)
```

Arguments

`X`	A matrix or data frame such that rows correspond to observations and columns correspond to variables. Note that this function currently only works with multivariate data p > 1.
`G`	A single positive integer value representing number of groups.
`model_id`	An integer representing the model_id, is useful for keeping track within parallizations. Not to be confused with model_type.
`model_type`	The type of covariance model you wish to run. Lexicon is given as follows: "0" = "EII", "1" = "VII", "2" = "EEI" , "3" = "EVI", "4" = "VEI", "5" = "VVI", "6" = "EEE", "7" = "VEE", "8" = "EVE", "9" = "EEV", "10" = "VVE", "11" = "EVV", "12" = "VEV", "13" = "VVV"
`in_zigs`	A n times G a posteriori matrix resembling the probability of observation i belonging to group G. Rows must sum to one, have the proper dimensions, and be positive.
`in_nmax`	Positive integer value resembling the maximum amount of iterations for the EM.
`in_l_tol`	A likelihood tolerance for convergence.
`in_m_iter_max`	For certain models, where applicable, the number of iterations for the maximization step.
`in_m_tol`	For certain models, where applicable, the tolerance for the maximization step.
`anneals`	A vector of doubles representing the deterministic annealing settings.
`t_burn`	A positive integer representing the number of burn steps if missing data (NAs) are detected.

Details

Be extremly careful running this function, it is known to crash systems without proper exception handling. Consider using the package `parallel` to estimate all possible models at the same time. Or run several possible initializations with random seeds.

Value

`zigs`	a postereori matrix
`G`	An integer representing the number of groups.
`sigs`	A vector of covariance matrices for each group (note you may have to reshape this)
`mus`	A vector of locational vectors for each group
`vgs`	Gamma parameters for each group

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

McNicholas, P.D. (2016), *Mixture Model-Based Classification*. Boca Raton: Chapman & Hall/CRC Press

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Andrews, J.L. and McNicholas, P.D. (2012), 'Model-based clustering, classification, and discriminant analysis via mixtures of multivariate t-distributions', Statistics and Computing 22(5), 1021-1029.

Examples

```
## Not run:

data("x2")
data_in = as.matrix(x2,ncol = 2)
n_iter = 300

in_g = 3
n = dim(data_in)[1]
model_string <- "VEI"
in_model_type <- switch(model_string, "EII" = 0,"VII" = 1,
              "EEI" = 2,  "EVI" = 3,  "VEI" = 4,  "VVI" = 5,  "EEE" = 6,
              "VEE" = 7,  "EVE" = 8,  "EEV" = 9,  "VVE" = 10,
              "EVV" = 11,"VEV" = 12,"VVV" = 13)

zigs_in <- z_ig_random_soft(n,in_g)

m2 = main_loop_t(X = data_in,
               G = 3, # number of groups
               model_id = 1, # model id for parallelization later
               model_type = in_model_type,
               in_zigs = zigs_in, # initializaiton
```

```
                in_nmax = n_iter, # number of iterations
                in_l_tol = 0.5, # likilihood tolerance
                in_m_iter_max = 20, # maximium iterations for matrices
                anneals=c(1),
                in_m_tol = 1e-8)

plot(x2,col = MAP(m2$zigs) + 1, cex = 0.5, pch = 20)

## End(Not run)
```

---

main_loop_vg *VGPCM Internal C++ Call*

Description

This function is the internal C++ function call within the vgpcm function. This is a raw C++ function call, meaning it has no checks for proper inputs so it may fail to run without giving proper errors. Please ensure all arguements are valid. main_loop_vg is useful for writing parallizations of the stpcm function. All arguement descriptions are given in terms of their corresponding C++ types.

Usage

```
main_loop_vg(X, G, model_id,
        model_type, in_zigs,
        in_nmax, in_l_tol, in_m_iter_max,
        in_m_tol, anneals,
        latent_step="standard",
        t_burn = 5L)
```

Arguments

X	A matrix or data frame such that rows correspond to observations and columns correspond to variables. Note that this function currently only works with multivariate data p > 1.
G	A single positive integer value representing number of groups.
model_id	An integer representing the model_id, is useful for keeping track within paralizations. Not to be confused with model_type.
model_type	The type of covariance model you wish to run. Lexicon is given as follows: "0" = "EII", "1" = "VII", "2" = "EEI" , "3" = "EVI", "4" = "VEI", "5" = "VVI", "6" = "EEE", "7" = "VEE", "8" = "EVE", "9" = "EEV", "10" = "VVE", "11" = "EVV", "12" = "VEV", "13" = "VVV"
in_zigs	A n times G a posteriori matrix resembling the probability of observation i belonging to group G. Rows must sum to one, have the proper dimensions, and be positive.
in_nmax	Positive integer value resembling the maximum amount of iterations for the EM.
in_l_tol	A likelihood tolerance for convergence.

`in_m_iter_max`	For certain models, where applicable, the number of iterations for the maximization step.
`in_m_tol`	For certain models, where applicable, the tolerance for the maximization step.
`anneals`	A vector of doubles representing the deterministic annealing settings.
`t_burn`	A positive integer representing the number of burn steps if missing data (NAs) are detected.
`latent_step`	If `"standard"`, it will use the standard E step for latent variable of a Normal Variance Mean Mixture, if `"random"` it will run a random draw from a GIG distribution.

Details

Be extremly careful running this function, it is known to crash systems without proper exception handling. Consider using the package `parallel` to estimate all possible models at the same time. Or run several possible initializations with random seeds.

Value

`zigs`	a postereori matrix
`G`	An integer representing the number of groups.
`sigs`	A vector of covariance matrices for each group (note you may have to reshape this)
`mus`	A vector of locational vectors for each group
`alphas`	A vector of skewness vectors for each group
`gammas`	Gamma parameters for each group

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

McNicholas, P.D. (2016), *Mixture Model-Based Classification*. Boca Raton: Chapman & Hall/CRC Press

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Zhou, H. and Lange, K. (2010). On the bumpy road to the dominant mode. *Scandinavian Journal of Statistics* **37**, 612-631.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Examples

```
## Not run:

data("sx2")
data_in = as.matrix(sx2,ncol = 2)
n_iter = 300

in_g = 2
n = dim(data_in)[1]
model_string <- "VVV"
in_model_type <- switch(model_string, "EII" = 0,"VII" = 1,
              "EEI" = 2,  "EVI" = 3,  "VEI" = 4,  "VVI" = 5,  "EEE" = 6,
              "VEE" = 7,  "EVE" = 8,  "EEV" = 9,  "VVE" = 10,
              "EVV" = 11,"VEV" = 12,"VVV" = 13)

zigs_in <- z_ig_random_soft(n,in_g)

m2 = main_loop_vg(X = t(data_in), # data in has to be in column major form
               G = 2, # number of groups
               model_id = 1, # model id for parallelization later
               model_type = in_model_type,
               in_zigs = zigs_in, # initializaiton
               in_nmax = n_iter, # number of iterations
               in_l_tol = 0.5, # likilihood tolerance
               in_m_iter_max = 20, # maximium iterations for matrices
               anneals=c(1),
               in_m_tol = 1e-8)

plot(sx2,col = MAP(m2$zigs) + 1, cex = 0.5, pch = 20)

## End(Not run)
```

MAP *Maximum* a posterori

Description

Generates labels from a classification matrix z

Usage

```
MAP(z_ig)
```

Arguments

z_ig	A classification matrix of positive numbers in which all rows must sum to one.

Value

A numeric matrix is returned of size n times g, with row sums adding up to 1.

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

Examples

```
  ## Not run:

# Simple example.
MAP(z_ig_random_soft(100,2))

# import dataset.
data(x2)
mm <- gpcm(data = as.matrix(x2),G = 1:7,
           start = 2,
           veo = FALSE,pprogress=FALSE)

best = get_best_model(mm)
# You can get labels using the internal object with MAP.
labs <- MAP(best$model_obj[[1]]$zigs)
# or you can just get labels directly.
labs2 <- best$map

## End(Not run)
```

`mixture` *Mixture Models for Clustering and Classification*

Description

An implementation of 14 parsimonious clustering models for finite mixtures with components that are Gaussian, generalized hyperbolic, variance-gamma, Student's t, or skew-t, for model-based clustering and model-based classification, even with missing data.

Details

Package:	mixture
Type:	Package
Version:	2.1.1
Date:	2024-01-29
License:	GPL (>=2)

This package contains the functions `gpcm`, `tpcm`, `ghpcm`, `vgpcm`, `stpcm`, `e_step`, `ARI`, and `get_best_model`, plus three simulated data sets.

This package also contains advanced functions for large system use which are: `main_loop` `main_loop_vg` , `main_loop_gh`, `main_loop_t` , `main_loop_st` ,`z_ig_random_soft`, `z_ig_random_hard`, `z_ig_kmeans`.

Author(s)

Nik Pocuca, Ryan P. Browne, and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

See Also

Details, examples, and references are given under gpcm, tpcm, ghpcm, stpcm, and vgpcm.

---

pcm *Parsimonious Clustering Models*

Description

Carries out model-based clustering or classification using some or all of the 14 parsimonious settings with any one of the GPCM,STPCM,VGPCM, or GHPCM families.

Usage

```
pcm(data=NULL, G=1:3, pcmfamily=c(gpcm,vgpcm,tpcm),
  mnames=NULL, start=2, label=NULL,
  veo=FALSE, da=c(1.0),
  nmax=1000, atol=1e-8, mtol=1e-8, mmax=10, burn=5,
  pprogress=FALSE, pwarning=FALSE, seed=123)
```

Arguments

data	A matrix or data frame such that rows correspond to observations and columns correspond to variables. Note that this function currently only works with multivariate data p > 1.
G	A sequence of integers giving the number of components to be used.
pcmfamily	The family of models to be used. If `NULL` then all are fitted.
mnames	The models (i.e., covariance structures) to be used. If `NULL` then all 14 are fitted.
start	If `0` then the random soft function is used for initialization. If `1` then the random hard function is used for initialization. If `2` then the kmeans function is used for initialization. If `>2` then multiple random soft starts are used for initialization. If `is.matrix` then matrix is used as an initialization matrix as along as it has non-negative elements. Note: only models with the same number of columns of this matrix will be fit.
label	If `NULL` then the data has no known groups. If `is.integer` then some of the observations have known groups. If `label[i]=k` then observation belongs to group `k`. If `label[i]=0` then observation has no known group. See Examples.
veo	Stands for "Variables exceed observations". If `TRUE` then if the number variables in the model exceeds the number of observations the model is still fitted.
da	Stands for Determinstic Annealing. A vector of doubles.

nmax	The maximum number of iterations each EM algorithm is allowed to use.
atol	A number specifying the epsilon value for the convergence criteria used in the EM algorithms. For each algorithm, the criterion is based on the difference between the log-likelihood at an iteration and an asymptotic estimate of the log-likelihood at that iteration. This asymptotic estimate is based on the Aitken acceleration and details are given in the References.
mtol	A number specifying the epsilon value for the convergence criteria used in the M-step in the EM algorithms.
mmax	The maximum number of iterations each M-step is allowed in the GEM algorithms.
burn	The burn in period for imputing data. (Missing observations are removed and a model is estimated seperately before placing an imputation step within the EM.)
pprogress	If TRUE print the progress of the function.
pwarning	If TRUE print the warnings.
seed	The seed for the run, default is 123

Details

The data x are either clustered or classified using Skew-t mixture models with some or all of the 14 parsimonious covariance structures described in Celeux & Govaert (1995). The algorithms given by Celeux & Govaert (1995) is used for 12 of the 14 models; the "EVE" and "VVE" models use the algorithms given in Browne & McNicholas (2014). Starting values are very important to the successful operation of these algorithms and so care must be taken in the interpretation of results.

Value

An object of class pcm is a list with components:

gpcm	If applicable, the output of running the Gaussian Parsimonious Family.
vgpcm	If applicable, the output of running the Variance-Gamma Parsimonious Family.
stpcm	If applicable, the output of running the Skew-T Parsimonious Family.
ghpcm	If applicable, the output of running the Generalized Hyperbolic Parsimonious Family.
best_model	An object of corresponding to the output of the best performing family.

Note

Dedicated print, and summary functions are available for objects of class pcm, gpcm, ghpcm, stpcm, or vgpcm.

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

McNicholas, P.D. (2016), *Mixture Model-Based Classification*. Boca Raton: Chapman & Hall/CRC Press

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Browne, R.P. and McNicholas, P.D. (2015), 'A mixture of generalized hyperbolic distributions', Canadian Journal of Statistics 43(2), 176-198.

Wei, Y., Tang, Y. and McNicholas, P.D. (2019), 'Mixtures of generalized hyperbolic distributions and mixtures of skew-t distributions for model-based clustering with incomplete data', Computational Statistics and Data Analysis 130, 18-41.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Examples

```
data("x2")

## Not run:

### estimate "VVV" "EVE"
ax = pcm(sx3, G=1:3, mnames=c("VVV","EVE"), start=0)
summary(ax)

print(ax)


## End(Not run)
```

---

stpcm *Skew-t Parsimonious Clustering Models*

Description

Carries out model-based clustering or classification using some or all of the 14 parsimonious Skew-t clustering models (STPCM).

Usage

```
stpcm(data=NULL, G=1:3, mnames=NULL,
  start=2, label=NULL,
  veo=FALSE, da=c(1.0),
  nmax=1000, atol=1e-8, mtol=1e-8, mmax=10, burn=5,
  pprogress=FALSE, pwarning=FALSE,
  stochastic = FALSE, latent_method="standard", seed=123)
```

Arguments

`data`	A matrix or data frame such that rows correspond to observations and columns correspond to variables. Note that this function currently only works with multivariate data p > 1.
`G`	A sequence of integers giving the number of components to be used.
`mnames`	The models (i.e., covariance structures) to be used. If `NULL` then all 14 are fitted.
`start`	If `0` then the random soft function is used for initialization. If `1` then the random hard function is used for initialization. If `2` then the kmeans function is used for initialization. If `>2` then multiple random soft starts are used for initialization. If `is.matrix` then matrix is used as an initialization matrix as along as it has non-negative elements. Note: only models with the same number of columns of this matrix will be fit.
`label`	If `NULL` then the data has no known groups. If `is.integer` then some of the observations have known groups. If `label[i]=k` then observation belongs to group `k`. If `label[i]=0` then observation has no known group. See Examples.
`veo`	Stands for "Variables exceed observations". If `TRUE` then if the number variables in the model exceeds the number of observations the model is still fitted.
`da`	Stands for Determinstic Annealing. A vector of doubles.
`nmax`	The maximum number of iterations each EM algorithm is allowed to use.
`atol`	A number specifying the epsilon value for the convergence criteria used in the EM algorithms. For each algorithm, the criterion is based on the difference between the log-likelihood at an iteration and an asymptotic estimate of the log-likelihood at that iteration. This asymptotic estimate is based on the Aitken acceleration and details are given in the References.
`mtol`	A number specifying the epsilon value for the convergence criteria used in the M-step in the EM algorithms.
`mmax`	The maximum number of iterations each M-step is allowed in the GEM algorithms.
`burn`	The burn in period for imputing data. (Missing observations are removed and a model is estimated seperately before placing an imputation step within the EM.)
`pprogress`	If `TRUE` print the progress of the function.
`pwarning`	If `TRUE` print the warnings.
`stochastic`	If `TRUE` , it will run stochastic E step variant.
`latent_method`	If `"standard"`, it will use the standard E step for latent variable of a Normal Variance Mean Mixture, if `"random"` it will run a random draw from a GIG distribution.
`seed`	The seed for the run, default is 123

Details

The data `x` are either clustered or classified using Skew-t mixture models with some or all of the 14 parsimonious covariance structures described in Celeux & Govaert (1995). The algorithms given by Celeux & Govaert (1995) is used for 12 of the 14 models; the "EVE" and "VVE" models use the algorithms given in Browne & McNicholas (2014). Starting values are very important to the successful operation of these algorithms and so care must be taken in the interpretation of results.

Value

An object of class vgpcm is a list with components:

map	A vector of integers indicating the maximum *a posteriori* classifications for the best model.
model_objs	A list of all estimated models with parameters returned from the C++ call.
best_model	A class of vgpcm_best containing; the number of groups for the best model, the covariance structure, and Bayesian Information Criterion (BIC) value.
loglik	The log-likelihood values from fitting the best model.
z	A matrix giving the raw values upon which map is based.
BIC	A G by mnames by 3 dimensional array with values pertaining to BIC calculations. (legacy)
gpar	A list object for each cluster pertaining to parameters. (legacy)
startobject	The type of object inputted into start.
row_tags	If there were NAs in the original dataset, a vector of indices referencing the row of the imputed vectors is given.

Best Model: An object of class stpcm_best is a list with components:

model_type	A string containing summarized information about the type of model estimated (Covariance structure and number of groups).
model_obj	An internal list containing all parameters returned from the C++ call.
BIC	Bayesian Index Criterion (positive scale, bigger is better).
loglik	Log liklihood from the estimated model.
nparam	Number of a parameters in the mode.
startobject	The type of object inputted into start.
G	An integer representing the number of groups.
cov_type	A string representing the type of covariance matrix (see 14 models).
status	Convergence status of EM algorithm according to Aitken's Acceleration
map	A vector of integers indicating the maximum *a posteriori* classifications for the best model.
row_tags	If there were NAs in the original dataset, a vector of indices referencing the row of the imputed vectors is given.

Internal Objects: All classes contain an internal list called model_obj or model_objs with the following components:

zigs	a posteori matrix
G	An integer representing the number of groups.
sigs	A vector of covariance matrices for each group
mus	A vector of location vectors for each group
alphas	A vector containg skewness vectors for each group
gammas	A vector containing estimated gamma parameters for each group

Note

Dedicated `print`, `plot` and `summary` functions are available for objects of class `vgpcm`.

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

McNicholas, P.D. (2016), *Mixture Model-Based Classification*. Boca Raton: Chapman & Hall/CRC Press

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Wei, Y., Tang, Y. and McNicholas, P.D. (2019), 'Mixtures of generalized hyperbolic distributions and mixtures of skew-t distributions for model-based clustering with incomplete data', Computational Statistics and Data Analysis 130, 18-41.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Examples

```
data("sx3")

## Not run:

### estimate "VVV" "EVE"
ax = stpcm(sx3, G=1:3, mnames=c("VVV","EVE"), start=0)
summary(ax)
ax


### estimate all 14 covariance structures
ax = stpcm(sx3, G=1:3, mnames=NULL, start=0)
summary(ax)
ax

### model based classification
sx3.label = c(rep(1,1000),rep(2,1000))
plot(sx3, col=sx3.label)
axl = stpcm(sx3, G=2, mnames=c("VVV", "EVE"), label=sx3.label)
summary(axl)


## End(Not run)
```

sx2 *Skewed Simulated Data 1*

Description

Simulated data, with two variables and two groups, used to illustrate `ghpcm`, `stpcm`, `vgpcm`.

Usage

```
data(sx2)
```

Format

A data frame with 2000 observations and 2 columns.

Source

These data were simulated using R.

sx3 *Skewed Simulated Data 2*

Description

Simulated data, with two variables and two groups, that are close together, used to illustrate `ghpcm`, `stpcm`, `vgpcm`.

Usage

```
data(sx3)
```

Format

A data frame with 2000 observations and 2 columns.

Source

These data were simulated using R.

tpcm *Student T Parsimonious Clustering Models*

Description

Carries out model-based clustering or classification using some or all of the 14 parsimonious Student T clustering models (TPCM).

Usage

```
tpcm(data=NULL, G=1:3, mnames=NULL,
  start=2, label=NULL,
  veo=FALSE, da=c(1.0),
  nmax=1000, atol=1e-8, mtol=1e-8, mmax=10, burn=5,
  pprogress=FALSE, pwarning=FALSE, stochastic=FALSE,
  constrained = FALSE, seed=123)
```

Arguments

`data`	A matrix or data frame such that rows correspond to observations and columns correspond to variables. Note that this function currently only works with multivariate data $p > 1$.
`G`	A sequence of integers giving the number of components to be used.
`mnames`	The models (i.e., covariance structures) to be used. If `NULL` then all 14 are fitted.
`start`	If `0` then the random soft function is used for initialization. If `1` then the random hard function is used for initialization. If `2` then the kmeans function is used for initialization. If `>2` then multiple random soft starts are used for initialization. If `is.matrix` then matrix is used as an initialization matrix as along as it has non-negative elements. Note: only models with the same number of columns of this matrix will be fit.
`label`	If `NULL` then the data has no known groups. If `is.integer` then some of the observations have known groups. If `label[i]=k` then observation belongs to group `k`. If `label[i]=0` then observation has no known group. See Examples.
`veo`	Stands for "Variables exceed observations". If `TRUE` then if the number variables in the model exceeds the number of observations the model is still fitted.
`da`	Stands for Determinstic Annealing. A vector of doubles.
`nmax`	The maximum number of iterations each EM algorithm is allowed to use.
`atol`	A number specifying the epsilon value for the convergence criteria used in the EM algorithms. For each algorithm, the criterion is based on the difference between the log-likelihood at an iteration and an asymptotic estimate of the log-likelihood at that iteration. This asymptotic estimate is based on the Aitken acceleration and details are given in the References.
`mtol`	A number specifying the epsilon value for the convergence criteria used in the M-step in the EM algorithms.

`mmax`	The maximum number of iterations each M-step is allowed in the GEM algorithms.
`burn`	The burn in period for imputing data. (Missing observations are removed and a model is estimated seperately before placing an imputation step within the EM.)
`pprogress`	If `TRUE` print the progress of the function.
`pwarning`	If `TRUE` print the warnings.
`stochastic`	If `TRUE` , it will run stochastic E step variant.
`constrained`	If `TRUE`, it will constrain the degrees of freedom for student-t to be the same for all clusters.
`seed`	The seed for the run, default is 123

Details

The data `x` are either clustered or classified using Skew-t mixture models with some or all of the 14 parsimonious covariance structures described in Celeux & Govaert (1995). The algorithms given by Celeux & Govaert (1995) is used for 12 of the 14 models; the "EVE" and "VVE" models use the algorithms given in Browne & McNicholas (2014). Starting values are very important to the successful operation of these algorithms and so care must be taken in the interpretation of results.

Value

An object of class `tpcm` is a list with components:

`map`	A vector of integers indicating the maximum *a posteriori* classifications for the best model.
`model_objs`	A list of all estimated models with parameters returned from the C++ call.
`best_model`	A class of vgpcm_best containing; the number of groups for the best model, the covariance structure, and Bayesian Information Criterion (BIC) value.
`loglik`	The log-likelihood values from fitting the best model.
`z`	A matrix giving the raw values upon which `map` is based.
`BIC`	A G by mnames by 3 dimensional array with values pertaining to BIC calculations. (legacy)
`gpar`	A list object for each cluster pertaining to parameters. (legacy)
`startobject`	The type of object inputted into `start`.
`row_tags`	If there were NAs in the original dataset, a vector of indices referencing the row of the imputed vectors is given.

Best Model: An object of class `stpcm_best` is a list with components:

`model_type`	A string containg summarized information about the type of model estimated (Covariance structure and number of groups).
`model_obj`	An internal list containing all parameters returned from the C++ call.
`BIC`	Bayesian Index Criterion (positive scale, bigger is better).
`loglik`	Log liklihood from the estimated model.

nparam	Number of a parameters in the mode.
startobject	The type of object inputted into `start`.
G	An integer representing the number of groups.
cov_type	A string representing the type of covariance matrix (see 14 models).
status	Convergence status of EM algorithm according to Aitken's Acceleration
map	A vector of integers indicating the maximum *a posteriori* classifications for the best model.
row_tags	If there were NAs in the original dataset, a vector of indices referencing the row of the imputed vectors is given.

Internal Objects: All classes contain an internal list called `model_obj` or `model_objs` with the following components:

zigs	a posteori matrix
G	An integer representing the number of groups.
sigs	A vector of covariance matrices for each group
mus	A vector of location vectors for each group
vgs	A vector containing estimated gamma parameters for each group

Note

Dedicated `print`, `plot` and `summary` functions are available for objects of class `vgpcm`.

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

McNicholas, P.D. (2016), *Mixture Model-Based Classification*. Boca Raton: Chapman & Hall/CRC Press

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Andrews, J.L. and McNicholas, P.D. (2012), 'Model-based clustering, classification, and discriminant analysis via mixtures of multivariate t-distributions', Statistics and Computing 22(5), 1021-1029.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Examples

```
data("x2")

## Not run:

### estimate "VVV" "EVE"
ax = tpcm(x2, G=1:3, mnames=c("VVV","EVE"), start=0)
summary(ax)
ax


### estimate all 14 covariance structures
ax = tpcm(x2, G=1:3, mnames=NULL, start=0)
summary(ax)
ax


## End(Not run)
```

---

vgpcm *Variance Gamma Parsimonious Clustering Models*

Description

Carries out model-based clustering or classification using some or all of the 14 parsimonious Variance Gamma clustering models (VGPCM).

Usage

```
vgpcm(data=NULL, G=1:3, mnames=NULL,
  start=2, label=NULL,
  veo=FALSE, da=c(1.0),
  nmax=1000, atol=1e-8, mtol=1e-8, mmax=10, burn=5,
  pprogress=FALSE, pwarning=FALSE,
  stochastic = FALSE, latent_method="standard", seed=123)
```

Arguments

data	A matrix or data frame such that rows correspond to observations and columns correspond to variables. Note that this function currently only works with multivariate data $p > 1$.
G	A sequence of integers giving the number of components to be used.
mnames	The models (i.e., covariance structures) to be used. If NULL then all 14 are fitted.
start	If 0 then the random soft function is used for initialization. If 1 then the random hard function is used for initialization. If 2 then the kmeans function is used for initialization. If >2 then multiple random soft starts are used for initialization.

If is.matrix then matrix is used as an initialization matrix as along as it has non-negative elements. Note: only models with the same number of columns of this matrix will be fit.

label
: If NULL then the data has no known groups. If is.integer then some of the observations have known groups. If label[i]=k then observation belongs to group k. If label[i]=0 then observation has no known group. See Examples.

veo
: Stands for "Variables exceed observations". If TRUE then if the number variables in the model exceeds the number of observations the model is still fitted.

da
: Stands for Determinstic Annealing. A vector of doubles.

nmax
: The maximum number of iterations each EM algorithm is allowed to use.

atol
: A number specifying the epsilon value for the convergence criteria used in the EM algorithms. For each algorithm, the criterion is based on the difference between the log-likelihood at an iteration and an asymptotic estimate of the log-likelihood at that iteration. This asymptotic estimate is based on the Aitken acceleration and details are given in the References.

mtol
: A number specifying the epsilon value for the convergence criteria used in the M-step in the EM algorithms.

mmax
: The maximum number of iterations each M-step is allowed in the GEM algorithms.

burn
: The burn in period for imputing data. (Missing observations are removed and a model is estimated seperately before placing an imputation step within the EM.)

pprogress
: If TRUE print the progress of the function.

pwarning
: If TRUE print the warnings.

stochastic
: If TRUE , it will run stochastic E step variant.

latent_method
: If "standard", it will use the standard E step for latent variable of a Normal Variance Mean Mixture, if "random" it will run a random draw from a GIG distribution.

seed
: The seed for the run, default is 123

Details

The data x are either clustered or classified using Variance Gamma mixture models with some or all of the 14 parsimonious covariance structures described in Celeux & Govaert (1995). The algorithms given by Celeux & Govaert (1995) is used for 12 of the 14 models; the "EVE" and "VVE" models use the algorithms given in Browne & McNicholas (2014). Starting values are very important to the successful operation of these algorithms and so care must be taken in the interpretation of results.

Value

An object of class vgpcm is a list with components:

map
: A vector of integers indicating the maximum *a posteriori* classifications for the best model.

model_objs
: A list of all estimated models with parameters returned from the C++ call.

`best_model`	A class of vgpcm_best containing; the number of groups for the best model, the covariance structure, and Bayesian Information Criterion (BIC) value.
`loglik`	The log-likelihood values from fitting the best model.
`z`	A matrix giving the raw values upon which `map` is based.
`BIC`	A G by mnames by 3 dimensional array with values pertaining to BIC calculations. (legacy)
`startobject`	The type of object inputted into `start`.
`gpar`	A list object for each cluster pertaining to parameters. (legacy)
`row_tags`	If there were NAs in the original dataset, a vector of indices referencing the row of the imputed vectors is given.

Best Model: An object of class `vgpcm_best` is a list with components:

`model_type`	A string containg summarized information about the type of model estimated (Covariance structure and number of groups).
`model_obj`	An internal list containing all parameters returned from the C++ call.
`BIC`	Bayesian Index Criterion (positive scale, bigger is better).
`loglik`	Log liklihood from the estimated model.
`nparam`	Number of a parameters in the mode.
`startobject`	The type of object inputted into `start`.
`G`	An integer representing the number of groups.
`cov_type`	A string representing the type of covariance matrix (see 14 models).
`status`	Convergence status of EM algorithm according to Aitken's Acceleration
`map`	A vector of integers indicating the maximum *a posteriori* classifications for the best model.
`row_tags`	If there were NAs in the original dataset, a vector of indices referencing the row of the imputed vectors is given.

Internal Objects: All classes contain an internal list called `model_obj` or `model_objs` with the following components:

`zigs`	a posteori matrix
`G`	An integer representing the number of groups.
`sigs`	A vector of covariance matrices for each group
`mus`	A vector of location vectors for each group
`alphas`	A vector containg skewness vectors for each group
`gammas`	A vector containing estimated gamma parameters for each group

Note

Dedicated `print`, `plot` and `summary` functions are available for objects of class `vgpcm`.

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

McNicholas, P.D. (2016), *Mixture Model-Based Classification*. Boca Raton: Chapman & Hall/CRC Press

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Examples

```
## Not run:

data("sx2")
### use kmeans to find starting values
ax0 = vgpcm(sx2, G=1:3, mnames=c("VVV", "EVE"),start=2, pprogress=TRUE, atol=1e-2)
summary(ax0)
ax0

### use random soft initializations.
ax6 = vgpcm(sx2, G=1:3, mnames=c("VVV", "EVE"),start= 0)
summary(ax6)
ax6

### use deterministic annealing for starting values
axDA = vgpcm(sx2, G=1:3, mnames=c("VVV", "EVE"), start=0,da=c(0.3,0.5,0.8,1.0))
summary(axDA)
axDA

### estimate all 14 covariance structures
ax = vgpcm(sx2, G=1:3, mnames=NULL, start=0)
summary(ax)
ax

### model based classification
sx2.label = c(rep(1,1000),rep(2,1000))
plot(sx2, col=sx2.label)
axl = vgpcm(sx2, G=2, mnames=c("VVV", "EVE"), label=sx2.label)
summary(axl)

## End(Not run)
```

`x2` *Simulated Data*

Description

Simulated data, with two variables with three groups, used to illustrate `gpcm`.

Usage

```
data(x2)
```

Format

A data frame with 300 observations and 2 columns.

Source

These data were simulated using `R`.

`z_ig_kmeans` *K-means Initialization*

Description

Generates an initialization matrix for a dataset X using k-means.

Usage

```
z_ig_kmeans(X,g)
```

Arguments

`X`	A matrix or data frame such that rows correspond to observations and columns correspond to variables. Note that this function currently only works with multivariate data p > 1. Note. NO NAS allowed.
`g`	An integer representing the number of groups.

Value

A numeric matrix is returned of size n times g, with row sums adding up to 1.

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Zhou, H. and Lange, K. (2010). On the bumpy road to the dominant mode. *Scandinavian Journal of Statistics* **37**, 612-631.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Examples

```
#data("x2")
#z_init <- z_ig_kmeans(x2,g=3)
```

`z_ig_random_hard` *Random Hard Initialization*

Description

Generates an initialization matrix of size n times g using random hard.

Usage

```
z_ig_random_hard(n,g)
```

Arguments

`n`	Number of rows, must be positive.
`g`	Number of columns, must be positive.

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Zhou, H. and Lange, K. (2010). On the bumpy road to the dominant mode. *Scandinavian Journal of Statistics* **37**, 612-631.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Examples

```
z_init <- z_ig_random_hard(100,3)
```

z_ig_random_soft *Random Soft Initialization*

Description

Generates an initialization matrix of size n times g using random soft.

Usage

```
z_ig_random_soft(n,g)
```

Arguments

n	Number of rows, must be positive.
g	Number of columns, must be positive.

Author(s)

Nik Pocuca, Ryan P. Browne and Paul D. McNicholas.

Maintainer: Paul D. McNicholas <mcnicholas@math.mcmaster.ca>

References

Browne, R.P. and McNicholas, P.D. (2014). Estimating common principal components in high dimensions. *Advances in Data Analysis and Classification* **8**(2), 217-226.

Zhou, H. and Lange, K. (2010). On the bumpy road to the dominant mode. *Scandinavian Journal of Statistics* **37**, 612-631.

Celeux, G., Govaert, G. (1995). Gaussian parsimonious clustering models. *Pattern Recognition* **28**(5), 781-793.

Examples

```
z_init <- z_ig_random_soft(100,3)
```

Index