Package ‘rrMixture’

April 8, 2022

Type Package
Title Reduced-Rank Mixture Models
Version 0.1-2
Date 2022-04-06
Description We implement full-ranked, rank-penalized, and adaptive nuclear norm penalized estimation methods using multivariate mixture models proposed by Kang, Chen, and Yao (2022+).
Depends R (>= 3.4.0)
Imports MASS, Rcpp (>= 1.0.8), Matrix, matrixcalc, gtools, utils
Suggests bayesm, rrpack, knitr, rmarkdown
VignetteBuilder knitr
LinkingTo Rcpp, RcppArmadillo
License GPL (>= 2)
Encoding UTF-8
RoxygenNote 7.1.1
NeedsCompilation yes
Author Suyeon Kang [aut, cre],
          Weixin Yao [aut],
          Kun Chen [aut]
Maintainer Suyeon Kang <skang062@ucr.edu>
Repository CRAN
Date/Publication 2022-04-08 03:30:02 UTC

R topics documented:

  initialize para .................................................. 2
  plot ................................................................. 3
  rrmix ............................................................... 5
  rrmix sim norm .................................................... 8
  rrMixture .......................................................... 10
  summary .......................................................... 10
  tune rrmix ........................................................ 11
Description

‘initialize.para’ is used to initialize parameter estimates.

Usage

initialize.para(K, X, Y, ind0 = NULL,
                seed = NULL, km.nstart = 20, kmscale = FALSE, n.init = 100,
                commonvar = FALSE)

Arguments

K number of mixture components.
X n by p design matrix where n is the number of observations and p is the number
    of predictors.
Y n by q response matrix where n is the number of observations and q is the number
    of responses.
ind0 vector of length n, specifying the initial assignment of the mixture membership
    of n observations when there is prior information on the membership. If
    ‘NULL’, K-means clustering technique is used to assign the membership for n
    observations. Default is ‘NULL’.
seed seed number for the reproducibility of results. Default is ‘NULL’.
km.nstart number of random sets considered to perform K-means clustering. Only used
    for K-means clustering. Default is 20.
kmscale logical value, indicating whether Y is scaled prior to K-means clustering. Only
    used for K-means clustering. Default is ‘FALSE’.
n.init number of initializations to try. Two methods for initial clustering are used:
    K-means and random clustering.
commonvar logical value, indicating the homogeneity assumption of variance-covariance
    matrices across K mixture components. Default is ‘FALSE’.

Value

para array of length K. It consists of K lists, each of which contains initial estimates
    of membership probability, coefficient matrix, and variance-covariance matrix.

Author(s)

Suyeon Kang, University of California, Riverside. <skang062@ucr.edu>; Weixin Yao, University
of California, Riverside, <weixin.yao@ucr.edu>; Kun Chen, University of Connecticut, <kun.chen@uconn.edu>. 
plot

Visualize rrmix Objects

Description

S3 methods visualizing results for some objects generated by rrmix and tune.rrmix.

Usage

```r
## S3 method for class 'rrmix'
plot(
  x,
  pch.L = 1,
  pch.F = 2,
  col.L = "red",
  col.F = "blue",
  lty.L = 1,
  lty.F = 1,
)```

Examples

```r
# Simulation 1: Two Components Case
K2mod <- rrmix.sim.norm(K = 2, n = 100, p = 5, q = 5, rho = .5, 
                        b = 1, shift = 1, r.star = c(1, 3), sigma = c(1, 1), 
                        pr = c(.5, .5), seed = 1215)
K2ini <- initialize.para(K = 2, X = K2mod$X, Y = K2mod$Y, 
                        seed = 100)

# Simulation 2: Four Components Case
K4mod <- rrmix.sim.norm(K = 4, n = 600, p = 15, q = 15, 
                        rho = .5, b = 1, shift = 1, r.star = c(1, 1, 3, 3), 
                        sigma = c(1, 1, 1, 1), pr = c(.25, .25, .25, .25), 
                        seed = 1215)
K4ini <- initialize.para(K = 4, X = K4mod$X, Y = K4mod$Y, 
                        seed = 100)
```

References


See Also

rrmix.sim.norm
type = "b",
...}

## S3 method for class 'tune.rrmix'
plot(
x,
metric = c("bic", "soft.class.err", "hard.class.err", "est.err", "pred.err"),
col = "blue",
main = NULL,
xlab = NULL,
ylab = NULL,
swapxy = FALSE,
transform.x = NULL,
transform.y = NULL,
transform.z = NULL,
color.palette = hsv_palette(),
nlevels = 20,
...}

Arguments

x an object of class rrmix or tune.rrmix.
pch.L symbol to use for displaying log-likelihood.
pch.F symbol to use for displaying penalized log-likelihood.
col.L color code or name for displaying log-likelihood.
col.F color code or name displaying penalized log-likelihood.
lty.L line type for displaying log-likelihood.
lty.F line type for displaying penalized log-likelihood.
type character indicating the type of plotting.
... Other arguments for future usage.
metric performance metric to use for finding best 'rrmix' model. 'soft.class.err', 'hard.class.err', 'est.err', and 'pred.err' can only be used when true parameter values are known.
col the color(s) of the surface facets. Transparent colors are ignored.
main main title.
xlab title for the x-axis.
ylab title for the y-axis.
swapxy if TRUE, the parameter axes are swaped (only used in case of two parameters).
transform.x, transform.y, transform.z functions to transform the parameters (x and y) and the error measures (z). Ignored if NULL.
color.palette color palette used in contour plot.
nlevels number of levels used in contour plot.
Description

‘rrmix’ is used to estimate parameters of reduced-rank mixture models in multivariate linear regression using the full-ranked, rank-penalized, and adaptive nuclear norm penalized estimators proposed by Kang et. al. (2022+).

Usage

```r
rrmix(K = 2, X, Y, est = c("FR", "RP", "ANNP"),
      lambda = 0, gamma = 2, ind0 = NULL, para0 = NULL, seed = NULL,
      kmscale = FALSE, km.nstart = 20, n.init = 100, commonvar = FALSE,
      maxiter = 1000, maxiter.int = 100, thres = 1e-05, thres.int = 1e-05,
      visible = FALSE, para.true = NULL, ind.true = NULL)
```

Arguments

- **K**: number of mixture components.
- **X**: n by p design matrix where n is the number of observations and p is the number of predictors.
- **Y**: n by q response matrix where n is the number of observations and q is the number of responses.
- **est**: character, specifying the estimation method. ‘FR’, ‘RP’, and ‘ANNP’ refers to as the full-ranked, rank-penalized, and adaptive nuclear norm penalized method, respectively.
- **lambda**: numerical value, specifying tuning parameter. Only used in the estimation method of ‘RP’ and ‘ANNP’. If 0, all estimation methods (‘FR’, ‘RP’, and ‘ANNP’) provide the same estimation results.
- **gamma**: numerical value, specifying additional tuning parameter, only used in the estimation method of ‘ANNP’. It must be nonnegative.
- **ind0**: vector of length n, specifying the initial assignment of the mixture membership of n observations when there is prior information on the membership. If ‘NULL’, K-means clustering technique is used to assign the membership for n observations. Default is ‘NULL’.
- **para0**: array of length K. It consists of K lists, each of which contains initial values of membership probability, coefficient matrix, and variance-covariance matrix.
- **seed**: seed number for the reproducibility of initialization results in the EM algorithm. Default is ‘NULL’.
- **kmscale**: logical value, indicating whether Y is scaled prior to K-means clustering for initialization. Default is ‘FALSE’.
- **km.nstart**: number of random sets considered to perform K-means clustering for initialization. Default is 20.
n.init number of initializations to try. Two methods for initial clustering are used: K-means and random clustering.

commonvar logical value, indicating the homogeneity assumption of variance-covariance matrices across K mixture components. Default is ‘FALSE’.

maxiter maximum number of iterations for external iterative algorithm, used in all estimation methods.

maxiter.int maximum number of iterations for internal iterative algorithm, only used in the estimation method of ‘ANNP’.

thres threshold value for external EM algorithm, used in all estimation methods. It controls the termination of the EM algorithm.

thres.int threshold value for internal iterative algorithm, only used in the estimation method of ‘ANNP’. It controls the termination of the internal algorithm.

visible logical value, indicating whether the outputs from each iteration are printed. Useful when the whole algorithm takes long. Default is ‘FALSE’.

para.true array of length K. It consists of K lists, each of which contains a coefficient matrix and its true rank. Only used when true models are known, e.g., in a simulation study.

ind.true vector of length n, specifying the true mixture membership for n observations. Only returned when the true models are known.

Value An object of class rrmix containing the fitted model, including:

call original function call.

seed seed number which is set for the initialization.

n.est vector of length K, specifying the estimated number of observations in each mixture components.

para array of length K. It consists of K lists, each of which contains final estimates of membership probability, coefficient matrix, and variance-covariance matrix.

est.rank vector of length K, specifying the estimated ranks of coefficient matrices.

npar number of parameters in the model, used to estimate the BIC.

n.iter number of iterations (external EM algorithm).

lambda tuning parameter for the estimation method of ’RP’ or ’ANNP’.

gamma tuning parameter for the estimation method of ’ANNP’.

ind vector of length n, specifying the estimated mixture membership for n observations.

ind.true vector of length n, specifying the true mixture membership for n observations. Only returned when the true models are known.

loglik log-likelihood of the final model.

penloglik penalized log-likelihood of the final model.

penalty penalty in the penalized log-likelihood of the final model.
bic BIC of the final model.

avg.nn.iter average number of iterations for internal iterative algorithm, only returned for the estimation method of 'ANNP'.

resmat matrix containing the information for each iteration of the EM algorithm, e.g., iteration number, log-likelihood, penalized log-likelihood, difference between penalized log-likelihood values from two consecutive iterations, and computing time.

class.err Soft and hard classification errors for mixture membership. Only returned when the true models are known.
est.err estimation error from the comparison between the estimated and true coefficient matrices. Only returned when the true models are known.
pred.err prediction error. Only returned when the true models are known.

Author(s)
Suyeon Kang, University of California, Riverside, <skang062@ucr.edu>; Weixin Yao, University of California, Riverside, <weixin.yao@ucr.edu>; Kun Chen, University of Connecticut, <kun.chen@uconn.edu>.

References

See Also
rrmix.sim.norm, initialize.para

Examples

library(rrMixture)

#-----------------------------------------------------------#
# Real Data Example: Tuna Data
#-----------------------------------------------------------#
require(bayesm)
data(tuna)
tunaY <- log(tuna[, c("MOVE1", "MOVE2", "MOVE3", "MOVE4",
"MOVE5", "MOVE6", "MOVE7")])
tunaX <- tuna[, c("NSALE1", "NSALE2", "NSALE3", "NSALE4",
"NSALE5", "NSALE6", "NSALE7",
"LPRICE1", "LPRICE2", "LPRICE3", "LPRICE4",
"LPRICE5", "LPRICE6", "LPRICE7")]
tunaX <- cbind(intercept = 1, tunaX)

tuna.rp <- rrmix(K = 2, X = tunaX, Y = tunaY, lambda = 3, est = "RP",
seed = 100, n.init = 100)
summary(tuna.rp)
plot(tuna.rp)
# Adaptive nuclear norm penalized estimation

tuna.anmp <- rrmix(K = 2, X = tunaX, Y = tunaY, lambda = 3, gamma = 2, est = "ANNP",
                   seed = 100, n.init = 100)
summary(tuna.anmp)
plot(tuna.anmp)

#-----------------------------------------------------------#
# Simulation: Two Components Case
#-----------------------------------------------------------#
# Simulation Data
K2mod <- rrmix.sim.norm(K = 2, n = 100, p = 5, q = 5, rho = .5,
                         b = 1, shift = 1, r.star = c(1, 3), sigma = c(1, 1),
                         pr = c(.5, .5), seed = 1215)

# Rank-penalized estimation
K2.rp <- rrmix(K = 2, X = K2mod$X, Y = K2mod$Y, lambda = 1,
               seed = 17, est = "RP", ind.true = K2mod$ind.true,
               para.true = K2mod$para.true, n.init = 100)
summary(K2.rp)
plot(K2.rp)

# Adaptive nuclear norm penalized estimation
K2.anmp <- rrmix(K = 2, X = K2mod$X, Y = K2mod$Y, lambda = 1,
                 seed = 17, est = "ANNP", ind.true = K2mod$ind.true,
                 para.true = K2mod$para.true, n.init = 100)
summary(K2.anmp)
plot(K2.anmp)

---

desc

Description

`rrmix.sim.norm` is used to create synthetic data from the multivariate normal distribution, which is used in a numerical study of Kang et. al. (2022+).

Usage

`rrmix.sim.norm(`
  K = 2,
  n = 100,
  p = 5,
  q = 5,
  rho = 0.5,
  b = 1,
  shift = 1,
r.mix.sim.norm

r.star = NULL,
sigma = NULL,
pr = NULL,
seed = NULL
}

Arguments

K number of mixture components.
n number of observations.
p number of predictors including an intercept.
q number of responses.
rho correlation between predictors used to make a design matrix.
b signal strength which controls the magnitude of coefficient matrices.
shift mean shift which measures how separate the mixture components are.
r.star vector of length K, specifying the true ranks of K coefficient matrices.
sigma vector of length K, specifying the noise strength of K multivariate normal distributions.
pr vector of length K, specifying the multinomial probabilities for the K mixture components.
seed seed number for the reproducibility of results. Default is ‘NULL’.

Value

X n by p design matrix.
Y n by q response matrix.
E p by q error matrix.
ind.true vector of length n, specifying the true mixture membership for n observations.
para.true array of length K. It consists of K lists, each of which contains a coefficient matrix and its true rank.

Author(s)

Suyeon Kang, University of California, Riverside, <skang062@ucr.edu>; Weixin Yao, University of California, Riverside, <weixin.yao@ucr.edu>; Kun Chen, University of Connecticut, <kun.chen@uconn.edu>.

References

Examples

#-----------------------------------------------------------#
# Simulation 1: Two Components Case
#-----------------------------------------------------------#
K2mod <- rrmix.sim.norm(K = 2, n = 100, p = 5, q = 5, rho = .5,
b = 1, shift = 1, r.star = c(1, 3), sigma = c(1, 1),
pr = c(.5, .5), seed = 1215)

#-----------------------------------------------------------#
# Simulation 2: Four Components Case
#-----------------------------------------------------------#
K4mod <- rrmix.sim.norm(K = 4, n = 600, p = 15, q = 15,
rho = .5, b = 1, shift = 1, r.star = c(1, 1, 3, 3),
sigma = c(1, 1, 1, 1), pr = c(.25, .25, .25, .25),
seed = 1215)

rrMixture

rrMixture: Reduced-Rank Mixture Models.

Description

The rrMixture package provides three important functions currently: rrmix, rrmix.sim.norm, and initialize.para.

summary

Summarize rrmix Objects

Description

S3 methods summarizing objects generated by rrmix and tune.rrmix.

Usage

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

## S3 method for class 'tune.rrmix'
summary(  
  object,  
  metric = c("bic", "soft.class.err", "hard.class.err", "est.err", "pred.err"),  
  ...  
)
tune.rrmix

Arguments

- **object**: Object generated from `rrmix` or `tune.rrmix`.
- **...**: Other arguments for future usage.
- **metric**: Performance metric to use for finding best ‘rrmix’ model. ‘soft.class.err’, ‘hard.class.err’, ‘est.err’, and ‘pred.err’ can only be used when true parameter values are known.

**tune.rrmix**

Reduced-rank mixture models with optimal tuning parameter(s)

Description

Reduced-rank mixture models with optimal tuning parameter(s)

Usage

```r
tune.rrmix(K = NULL, K.max = NULL, X, Y, est = c("FR", "RP", "ANNP"),
  lambda = NULL, n.lambda = 20, gamma = 2,
  ind0 = NULL, para0 = NULL, seed = NULL, kmscale = FALSE, km.nstart = 20,
  n.init = 100, commonvar = FALSE, maxiter = 1000, maxiter.int = 100,
  thres = 1e-05, thres.int = 1e-05,
  para.true = NULL, ind.true = NULL)
```

Arguments

- **K**: Number of mixture components. Required when `K.max` is ‘NULL’.
- **K.max**: Maximum of mixture components. Default is ‘NULL’. When provided, the argument `K` is ignored.
- **X**: n by p design matrix where n is the number of observations and p is the number of predictors.
- **Y**: n by q response matrix where n is the number of observations and q is the number of responses.
- **est**: Character, specifying the estimation method. ‘FR’, ‘RP’, and ‘ANNP’ refers to as the full-ranked, rank-penalized, and adaptive nuclear norm penalized method, respectively.
- **lambda**: Vector consisting of lambda candidates. Only used in the estimation method of ‘RP’ and ‘ANNP’. If 0, all estimation methods (‘FR’, ‘RP’, and ‘ANNP’) provide the same estimation results. Default is ‘NULL’. If ‘NULL’, data-adaptive range of lambda will be provided internally.
- **n.lambda**: Number of lambda candidates to explore. Only used when ‘lambda’ is ‘NULL’. Default is 20.
- **gamma**: Numerical value, specifying additional tuning parameter, only used in the estimation method of ‘ANNP’. It must be nonnegative.
**ind0**
vector of length n, specifying the initial assignment of the mixture membership of n observations when there is prior information on the membership. If ‘NULL’, K-means clustering technique is used to assign the membership for n observations. Default is ‘NULL’.

**para0**
array of length K. It consists of K lists, each of which contains initial values of membership probability, coefficient matrix, and variance-covariance matrix.

**seed**
seed number for the reproducibility of results. Default of ‘NULL’.

**kmscale**
logical value, indicating whether Y is scaled prior to K-means clustering for initialization. Default is ‘FALSE’.

**km.nstart**
number of random sets considered to perform K-means clustering for initialization. Default is 20.

**n.init**
number of initializations to try. Two methods for initial clustering are used: K-means and random clustering.

**commonvar**
logical value, indicating the homogeneity assumption of variance-covariance matrices across K mixture components. Default is ‘FALSE’.

**maxiter**
maximum number of iterations for external iterative algorithm, used in all estimation methods.

**maxiter.int**
maximum number of iterations for internal iterative algorithm, only used in the estimation method of ‘ANNP’.

**thres**
threshold value for external EM algorithm, used in all estimation methods. It controls the termination of the EM algorithm.

**thres.int**
threshold value for internal iterative algorithm, only used in the estimation method of ‘ANNP’. It controls the termination of the internal algorithm.

**para.true**
array of length K. It consists of K lists, each of which contains a coefficient matrix and its true rank. Only used when true models are known, e.g., in a simulation study.

**ind.true**
vector of length n, specifying the true mixture membership for n observations. Only used when true models are known, e.g., in a simulation study.

**Value**

**lambda.cand**
lambda values used as input.

**penloglik**
penalized log-likelihood values corresponding to the set of lambda values.

**bic**
BIC values corresponding to the set of lambda values.

**est.rank**
estimated ranks corresponding to the set of lambda values.

**Author(s)**

Suyeon Kang, University of California, Riverside, <skang062@ucr.edu>; Weixin Yao, University of California, Riverside, <weixin.yao@ucr.edu>; Kun Chen, University of Connecticut, <kun.chen@uconn.edu>.

**References**

tune.rrmix

See Also

rrmix

Examples

#-----------------------------------------------------------#
# Real Data Example: Tuna Data                             #
#-----------------------------------------------------------#
require(bayesm)
data(tuna)
tunaY <- log(tuna[, c("MOVE1", "MOVE2", "MOVE3", "MOVE4",
                     "MOVE5", "MOVE6", "MOVE7"))
tunaX <- tuna[, c("NSALE1", "NSALE2", "NSALE3", "NSALE4",
                  "NSALE5", "NSALE6", "NSALE7",
                  "LPRICE1", "LPRICE2", "LPRICE3", "LPRICE4",
                  "LPRICE5", "LPRICE6", "LPRICE7")]
tunaX <- cbind(intercept = 1, tunaX)

#-----------------------------------------------------------#
# Tuna Example                                             #
#-----------------------------------------------------------#
tuna.tune <- tune.rrmix(K.max = 3, X = tunaX, Y = tunaY, est = "RP",
                        lambda = exp(seq(0, log(100), length = 20)),
                        seed = 100, n.init = 100)
supply(tuna.tune)
plot(tuna.tune, transform.y = log, ylab = "log(lambda)"
Index

initialize.para, 2, 7
plot, 3
rrmix, 5, 13
rrmix.sim.norm, 3, 7, 8
rrMixture, 10
summary, 10
tune.rrmix, 11