Package ‘rrpack’

October 14, 2022

Title Reduced-Rank Regression

Version 0.1-13

Description Multivariate regression methodologies including classical reduced-rank regression (RRR) studied by Anderson (1951) <doi:10.1214/aoms/1177729580> and Reinsel and Velu (1998) <doi:10.1007/978-1-4757-2853-8>, reduced-rank regression via adaptive nuclear norm penalization proposed by Chen et al. (2013) <doi:10.1093/biomet/ast036> and Mukherjee et al. (2015) <doi:10.1093/biomet/asx080>, robust reduced-rank regression (R4) proposed by She and Chen (2017) <doi:10.1093/biomet/asx032>, generalized/mixed-response reduced-rank regression (mRRR) proposed by Luo et al. (2018) <doi:10.1016/j.jmva.2018.04.011>, row-sparse reduced-rank regression (SRRR) proposed by Chen and Huang (2012) <doi:10.1080/01621459.2012.734178>, reduced-rank regression with a sparse singular value decomposition (RSSVD) proposed by Chen et al. (2012) <doi:10.1111/j.1467-9868.2011.01002.x> and sparse and orthogonal factor regression (SOFAR) proposed by Uematsu et al. (2019) <doi:10.1109/TIT.2019.2909889>.

Depends R (>= 3.4.0)

Imports ggplot2, glmnet, MASS, Rcpp (>= 0.12.0)

LinkingTo Rcpp, RcppArmadillo

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.0

NeedsCompilation yes

Author Kun Chen [aut, cre] (<https://orcid.org/0000-0003-3579-5467>),
Wenjie Wang [aut] (<https://orcid.org/0000-0003-0363-3180>),
Jun Yan [ctb] (<https://orcid.org/0000-0003-4401-7296>)

Maintainer Kun Chen <kun.chen@uconn.edu>

Repository CRAN

Date/Publication 2022-06-16 06:40:02 UTC

R topics documented:

`cv.mrrr` *Mixed-response reduced-rank regression with rank selected by cross validation*

Description

Mixed-response reduced-rank regression with rank selected by cross validation

Usage

```
cv.mrrr(
  Y,
  X,
  is.pca = NULL,
  offset = NULL,
  ctrl.id = c(),
  family = list(gaussian(), binomial(), poisson()),
  familygroup = NULL,
  maxrank = min(ncol(Y), ncol(X)),
  penstr = list(),
  init = list(),
```

```
  control = list(),
  nfold = 5,
  foldid = NULL,
  nlam = 20,
  warm = FALSE
)
```

Arguments

`Y`	response matrix
`X`	covariate matrix
`is.pca`	If TRUE, mixed principal component analysis with X=I
`offset`	matrix of the same dimension as Y for offset
`ctrl.id`	indices of unpenalized predictors
`family`	a list of family functions as used in `glm`
`familygroup`	a list of family indices of the responses
`maxrank`	integer giving the maximum rank allowed.
`penstr`	a list of penalty structure of SVD.
`init`	a list of initial values of kappaC0, kappaS0, C0, and S0
`control`	a list of controling parameters for the fitting
`nfold`	number of folds in cross validation
`foldid`	to specify the folds if desired
`nlam`	number of tuning parameters; not effective when using rank constrained estimation
`warm`	if TRUE, use warm start in fitting the solution paths

Value

S3 `mrrr` object, a list containing

`fit`	the output from the selected model
`dev`	deviance measures

Examples

```
## Not run:
library(rrpack)
simdata <- rrr.sim3(n = 100, p = 30, q.mix = c(5, 20, 5),
                    nrank = 2, mis.prop = 0.2)
Y <- simdata$Y
Y_mis <- simdata$Y.mis
X <- simdata$X
X0 <- cbind(1,X)
C <- simdata$C
family <- simdata$family
familygroup <- simdata$familygroup
```

```
svdX0d1 <- svd(X0)$d[1]
init1 = list(kappaC0 = svdX0d1 * 5)
offset = NULL
control = list(epsilon = 1e-4, sv.tol = 1e-2, maxit = 2000,
               trace = FALSE, gammaC0 = 1.1, plot.cv = TRUE,
               conv.obj = TRUE)
fit.cv.mrrr <- cv.mrrr(Y_mis, X, family = family,
                       familygroup = familygroup,
                       maxrank = 20,
                       penstr = list(penaltySVD = "rankCon",
                                     lambdaSVD = c(1 : 6)),
                       control = control, init = init1,
                       nfold = 10, nlam = 50)
summary(fit.cv.mrrr)
coef(fit.cv.mrrr)
fit.mrrr <- fit.cv.mrrr$fit

## plot(svd(fit.mrrr$coef[- 1,])$d)
plot(C ~ fit.mrrr$coef[- 1, ])
abline(a = 0, b = 1)

## End(Not run)
```

cv.rrr *Reduced-rank regression with rank selected by cross validation*

Description

Reduced-rank regression with rank selected by cross validation

Usage

```
cv.rrr(
  Y,
  X,
  nfold = 10,
  maxrank = min(dim(Y), dim(X)),
  norder = NULL,
  coefSVD = FALSE
)
```

Arguments

Y	response matrix
X	covariate matrix
nfold	number of folds
maxrank	maximum rank allowed
norder	for constructing the folds
coefSVD	If TRUE, svd of the coefficient is retuned

Value

a list containing rr estimates from cross validation

References

Chen, K., Dong, H. and Chan, K.-S. (2013) Reduced rank regression via adaptive nuclear norm penalization. *Biometrika*, 100, 901–920.

Examples

```
library(rrpack)
p <- 50; q <- 50; n <- 100; nrank <- 3
mydata <- rrr.sim1(n, p, q, nrank, s2n = 1, sigma = NULL,
                  rho_X = 0.5, rho_E = 0.3)
rfit_cv <- with(mydata, cv.rrr(Y, X, nfold = 10, maxrank = 10))
summary(rfit_cv)
coef(rfit_cv)
```

---

cv.sofar — *Sparse orthognal factor regression tuned by cross validation*

Description

Sparse orthognal factor regression tuned by cross validation

Usage

```
cv.sofar(
  Y,
  X,
  nrank = 1,
  su = NULL,
  sv = NULL,
  nfold = 5,
  norder = NULL,
  modstr = list(),
  control = list(),
  screening = FALSE
)
```

Arguments

Y	response matrix
X	covariate matrix
nrank	an integer specifying the desired rank/number of factors
su	a scaling vector for U such that $U^T U = diag(s_u)$

`sv`	a scaling vector for V such that $V^T V = diag(s_v)$
`nfold`	number of fold; used for cv.sofar
`norder`	observation orders to constrct data folds; used for cv.sofar
`modstr`	a list of internal model parameters controlling the model fitting
`control`	a list of internal computation parameters controlling optimization
`screening`	If TRUE, marginal screening via lasso is performed before sofar fitting.

Details

The model parameters can be specified through argument `modstr`. The available elements include

- mu: parameter in the augmented Lagrangian function.
- mugamma: increament of mu along iterations to speed up computation.
- WA: weight matrix for A.
- WB: weight matrix for B.
- Wd: weight matrix for d.
- wgamma: power parameter in constructing adaptive weights.

The model fitting can be controled through argument `control`. The avilable elements include

- nlam: number of lambda triplets to be used.
- lam.min.factor: set the smallest lambda triplets as a fraction of the estimation lambda.max triplets.
- lam.max.factor: set the largest lambda triplets as a multiple of the estimation lambda.max triplets.
- lam.AB.factor: set the relative penalty level between A/B and D.
- penA,penB,penD: if TRUE, penalty is applied.
- lamA: sequence of tuning parameters for A.
- lamB: sequence of tuning parameters for B.
- lamD: sequence of tuning parameters for d.
- methodA: penalty for penalizing A.
- methodB: penalty for penalizing B.
- epsilon: convergence tolerance.
- maxit: maximum number of iterations.
- innerEpsilon: convergence tolerance for inner subroutines.
- innerMaxit: maximum number of iterations for inner subroutines.
- sv.tol: tolerance for singular values.

cv.srrr *Row-sparse reduced-rank regression tuned by cross validation*

Description

Row-sparse reduced-rank regression tuned by cross validation

Usage

```
cv.srrr(
  Y,
  X,
  nrank = 1,
  method = c("glasso", "adglasso"),
  nfold = 5,
  norder = NULL,
  A0 = NULL,
  V0 = NULL,
  modstr = list(),
  control = list()
)
```

Arguments

`Y`	response matrix
`X`	covariate matrix
`nrank`	prespecified rank
`method`	group lasso or adaptive group lasso
`nfold`	fold number
`norder`	for constructing the folds
`A0`	initial value
`V0`	initial value
`modstr`	a list of model parameters controlling the model fitting
`control`	a list of parameters for controlling the fitting process

Details

Model parameters controlling the model fitting can be specified through argument `modstr`. The available elements include

- lamA: tuning parameter sequence.
- nlam: number of tuning parameters; no effect if `lamA` is specified.
- minLambda: minimum lambda value, no effect if `lamA` is specified.
- maxLambda: maxmum lambda value, no effect if lamA is specified.

- WA: adaptive weights. If NULL, the weights are constructed from RRR.
- wgamma: power parameter for constructing adaptive weights.

Similarly, the computational parameters controlling optimization can be specified through argument `control`. The available elements include

- epsilon: epsilonergence tolerance.
- maxit: maximum number of iterations.
- inner.eps: used in inner loop.
- inner.maxit: used in inner loop.

Value

A list of fitting results

References

Chen, L. and Huang, J.Z. (2012) Sparse reduced-rank regression for simultaneous dimension reduction and variable selection. *Journal of the American Statistical Association*. 107:500, 1533–1545.

---

mrrr — *Generalized or mixed-response reduced-rank regression*

Description

Peforms either rank constrained maximum likelihood estimation or singular value penalized estimation.

Usage

```
mrrr(
  Y,
  X,
  is.pca = NULL,
  offset = NULL,
  ctrl.id = c(),
  family = list(gaussian(), binomial()),
  familygroup = NULL,
  maxrank = min(ncol(Y), ncol(X)),
  penstr = list(),
  init = list(),
  control = list()
)
```

Arguments

`Y`	response matrix
`X`	covariate matrix
`is.pca`	If TRUE, mixed principal component analysis with X=I
`offset`	matrix of the same dimension as Y for offset
`ctrl.id`	indices of unpenalized predictors
`family`	a list of family functions as used in `glm`
`familygroup`	a list of family indices of the responses
`maxrank`	integer giving the maximum rank allowed. Usually this can be set to min(n,p,q)
`penstr`	a list of penalty structure of SVD, contains penstr$penaltySVD is the penalty of SVD, penstr$lambdaSVD is the regularization parameter
`init`	a list of initial values of kappaC0, kappaS0, C0, and S0
`control`	a list of controling parameters for the fitting

Details

The model fitting process can be fine tuned through argument `control`. The available elements for `control` include

- epsilon: positive convergence tolerance epsilon; the iterations converge when |new - old | / (old + 0.1) < epsilon. treated as zero.
- sv.tol: tolerance for singular values.
- maxit: integer giving the maximal number of iterations.
- trace:logical indicating if tracing the objective is needed.
- conv.obj:if TRUE, track objective function.
- equal.phi:if TRUE, use a single dispersion parameter for Gaussian responses.
- plot.obj:if TRUE, plot obj values along iterations; for checking only
- plot.cv:if TRUE, plot cross validation error.
- gammaC0:adaptive scaling to speed up computation.

Similarly, the available elements for arguments `penstr` specifying penalty structure of SVD include

- penaltySVD: penalty for reducing rank
- lambdaSVD: tuning parameter. For penaltySVD = rankCon, this is the specified rank.

Value

S3 `mrrr` object, a list containing

`obj`	the objective function tracking
`converged`	TRUE/FALSE for convergence
`coef`	the estimated coefficient matrix
`outlier`	the estimated outlier matrix
`nrank`	the rank of the fitted model

Examples

```
library(rrpack)
simdata <- rrr.sim3(n = 100, p = 30, q.mix = c(5, 20, 5),
                    nrank = 2, mis.prop = 0.2)
Y <- simdata$Y
Y_mis <- simdata$Y.mis
X <- simdata$X
X0 <- cbind(1, X)
C <- simdata$C
family <- simdata$family
familygroup <- simdata$familygroup
svdX0d1 <- svd(X0)$d[1]
init1 = list(kappaC0 = svdX0d1 * 5)
offset = NULL
control = list(epsilon = 1e-4, sv.tol = 1e-2, maxit = 2000,
               trace = FALSE, gammaC0 = 1.1, plot.cv = TRUE,
               conv.obj = TRUE)
fit.mrrr <- mrrr(Y_mis, X, family = family, familygroup = familygroup,
                 penstr = list(penaltySVD = "rankCon", lambdaSVD = 2),
                 control = control, init = init1)
summary(fit.mrrr)
coef(fit.mrrr)
par(mfrow = c(1, 2))
plot(fit.mrrr$obj)
plot(C ~ fit.mrrr$coef[- 1 ,])
abline(a = 0, b = 1)
```

---

plot *Scatter Plot*

Description

S3 methods generating scatter plot for some objects generated by rrpack using ggplot2. An ggplot2 object is returned so that users are allowed to easily further customize the plot.

Usage

```
## S3 method for class 'rrr'
plot(
  x,
  y = NULL,
  layer = 1L,
  xlab = paste("latent predictor ", layer, sep = ""),
  ylab = paste("latent response ", layer, sep = ""),
  ...
)

## S3 method for class 'sofar'
plot(
```

```
  x,
  y = NULL,
  layer = 1L,
  xlab = paste("latent predictor ", layer, sep = ""),
  ylab = paste("latent response ", layer, sep = ""),
  ...
)

## S3 method for class 'cv.sofar'
plot(
  x,
  y = NULL,
  layer = 1L,
  xlab = paste("latent predictor ", layer, sep = ""),
  ylab = paste("latent response ", layer, sep = ""),
  ...
)

## S3 method for class 'srrr'
plot(
  x,
  y = NULL,
  layer = 1L,
  xlab = paste("latent predictor ", layer, sep = ""),
  ylab = paste("latent response ", layer, sep = ""),
  ...
)

## S3 method for class 'cv.srrr'
plot(
  x,
  y = NULL,
  layer = 1L,
  xlab = paste("latent predictor ", layer, sep = ""),
  ylab = paste("latent response ", layer, sep = ""),
  ...
)

## S3 method for class 'rssvd'
plot(
  x,
  y = NULL,
  layer = 1L,
  xlab = paste("latent predictor ", layer, sep = ""),
  ylab = paste("latent response ", layer, sep = ""),
  ...
)
```

Arguments

`x`	Some object generated by `rrpack`.
`y`	NULL. Do not need to specify.
`layer`	The unit-rank layer to plot; cannot be larger than the estimated rank
`xlab`	Label of X axis.
`ylab`	Label of Y axis.
`...`	Other argumnts for future usage.

Value

ggplot2 object.

`r4` *Robust reduced-rank regression*

Description

Perform robust reduced-rank regression.

Usage

```
r4(
  Y,
  X,
  maxrank = min(dim(Y), dim(X)),
  method = c("rowl0", "rowl1", "entrywise"),
  Gamma = NULL,
  ic.type = c("AIC", "BIC", "PIC"),
  modstr = list(),
  control = list()
)
```

Arguments

`Y`	a matrix of response (n by q)
`X`	a matrix of covariate (n by p)
`maxrank`	maximum rank for fitting
`method`	outlier detection method, either entrywise or rowwise
`Gamma`	weighting matrix in the loss function
`ic.type`	information criterion, AIC, BIC or PIC
`modstr`	a list of model parameters controlling the model fitting
`control`	a list of parameters for controlling the fitting process

Details

The model parameters can be controlled through argument `modstr`. The available elements include

- nlam: parameter in the augmented Lagrangian function.
- adaptive: if `TRUE`, use leverage values for adaptive penalization. The default value is `FALSE`.
- weights: user supplied weights for adaptive penalization.
- minlam: maximum proportion of outliers.
- maxlam: maximum proportion of good observations.
- delid: discarded observation indices for initial estimation.

The model fitting can be controlled through argument `control`. The available elements include

- epsilon: convergence tolerance.
- maxit: maximum number of iterations.
- qr.tol: tolerance for qr decomposition.
- tol: tolerance.

Value

a list consisting of

`coef.path`	solutuon path of regression coefficients
`s.path`	solutuon path of sparse mean shifts
`s.norm.path`	solutuon path of the norms of sparse mean shifts
`ic.path`	paths of information criteria
`ic.smooth.path`	smoothed paths of information criteria
`lambda.path`	paths of the tuning parameter
`id.solution`	ids of the selected solutions on the path
`ic.best`	lowest values of the information criteria
`rank.best`	rank values of selected solutions
`coef`	estimated regression coefficients
`s`	estimated sparse mean shifts
`rank`	rank estimate

References

She, Y. and Chen, K. (2017) Robust reduced-rank regression. *Biometrika*, 104 (3), 633–647.

Examples

```
## Not run:
library(rrpack)
n <- 100; p <- 500; q <- 50
xrank <- 10; nrank <- 3; rmax <- min(n, p, q, xrank)
nlam <- 100; gamma <- 2
rho_E <- 0.3
rho_X <- 0.5
nlev <- 0
vlev <- 0
vout <- NULL
vlevsd <- NULL
nout <- 0.1 * n
s2n <- 1
voutsd <- 2
simdata <- rrr.sim5(n, p, q, nrank, rx = xrank, s2n = s2n,
                    rho_X = rho_X, rho_E = rho_E, nout = nout, vout = vout,
                    voutsd = voutsd,nlev = nlev,vlev=vlev,vlevsd=vlevsd)
Y <- simdata$Y
X <- simdata$X
fit <- r4(Y, X, maxrank = rmax,
               method = "rowl0", ic.type= "PIC")
summary(fit)
coef(fit)
which(apply(fit$s,1,function(a)sum(a^2))!=0)

## End(Not run)
```

`rrpack-coef` *Estimated coefficients*

Description

S3 methods extracting estimated coefficients for objects generated by `rrpack`.

Usage

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

## S3 method for class 'cv.mrrr'
coef(object, ...)

## S3 method for class 'r4'
coef(object, ...)

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

```
## S3 method for class 'rrr.fit'
coef(object, ...)

## S3 method for class 'cv.rrr'
coef(object, ...)

## S3 method for class 'srrr'
coef(object, ...)

## S3 method for class 'sofar'
coef(object, ...)

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

Arguments

`object`	Object generated by `rrpack`.
`...`	Other argumnts for future usage.

Value

A numeric matrix.

`rrr` *Multivariate reduced-rank linear regression*

Description

Produce solution paths of reduced-rank estimators and adaptive nuclear norm penalized estimators; compute the degrees of freeom of the RRR estimators and select a solution via certain information criterion.

Usage

```
rrr(
  Y,
  X,
  penaltySVD = c("rank", "ann"),
  ic.type = c("GIC", "AIC", "BIC", "BICP", "GCV"),
  df.type = c("exact", "naive"),
  maxrank = min(dim(Y), dim(X)),
  modstr = list(),
  control = list()
)
```

Arguments

`Y`	a matrix of response (n by q)
`X`	a matrix of covariate (n by p)
`penaltySVD`	'rank': rank-constrainted estimation; 'ann': adaptive nuclear norm estimation.
`ic.type`	the information criterion to be used; currently supporting 'AIC', 'BIC', 'BICP', 'GCV', and 'GIC'.
`df.type`	'exact': the exact degrees of freedoms based on SURE theory; 'naive': the naive degress of freedoms based on counting number of free parameters
`maxrank`	an integer of maximum desired rank.
`modstr`	a list of model parameters controlling the model fitting
`control`	a list of parameters for controlling the fitting process: 'sv.tol' controls the tolerence of singular values; 'qr.tol' controls the tolerence of QR decomposition for the LS fit

Details

Model parameters can be specified through argument `modstr`. The available include

- gamma: A scalar power parameter of the adaptive weights in `penalty == "ann"`.
- nlambda: The number of lambda values; no effect if `penalty == "count"`.
- lambda: A vector of user-specified rank values if `penalty == "count"` or a vector of penalty values if `penalty == "ann"`.

The available elements for argument `control` include

- sv.tol: singular value tolerence.
- qr.tol: QR decomposition tolerence.

Value

S3 `rrr` object, a list consisting of

`call`	original function call
`Y`	input matrix of response
`X`	input matrix of covariate
`A`	right singular matri x of the least square fitted matrix
`Ad`	a vector of squared singular values of the least square fitted matrix
`coef.ls`	coefficient estimate from LS
`Spath`	a matrix, each column containing shrinkage factors of the singular values of a solution; the first four objects can be used to recover all reduced-rank solutions
`df.exact`	the exact degrees of freedom
`df.naive`	the naive degrees of freedom
`penaltySVD`	the method of low-rank estimation
`sse`	a vecotr of sum of squard errors

ic	a vector of information criterion
coef	estimated coefficient matrix
U	estimated left singular matrix such that XU/sqrtn is orthogonal
V	estimated right singular matrix that is orthogonal
D	estimated singular value matrix such that C = UDVt
rank	estimated rank

References

Chen, K., Dong, H. and Chan, K.-S. (2013) Reduced rank regression via adaptive nuclear norm penalization. *Biometrika*, 100, 901–920.

Examples

```
library(rrpack)
p <- 50; q <- 50; n <- 100; nrank <- 3
mydata <- rrr.sim1(n, p, q, nrank, s2n = 1, sigma = NULL,
                   rho_X = 0.5, rho_E = 0.3)
rfit <- with(mydata, rrr(Y, X, maxrank = 10))
summary(rfit)
coef(rfit)
plot(rfit)
```

---

rrr.cookD *Cook's distance in reduced-rank regression for model diagnostics*

Description

Compute Cook's distance for model diagnostics in `rrr` estimation.

Usage

```
rrr.cookD(Y, X = NULL, nrank = 1, qr.tol = 1e-07)
```

Arguments

Y	response matrix
X	covariate matrix
nrank	model rank
qr.tol	tolerance

Value

a list containing diagnostics measures

References

Chen, K. Model diagnostics in reduced-rank estimation. *Statistics and Its interface*, 9, 469–484.

`rrr.fit` *Fitting reduced-rank regression with a specific rank*

Description

Given a response matrix and a covariate matrix, this function fits reduced rank regression for a specified rank. It reduces to singular value decomposition if the covariate matrix is the identity matrix.

Usage

```
rrr.fit(Y, X, nrank = 1, weight = NULL, coefSVD = FALSE)
```

Arguments

`Y`	a matrix of response (n by q)
`X`	a matrix of covariate (n by p)
`nrank`	an integer specifying the desired rank
`weight`	a square matrix of weight (q by q); The default is the identity matrix
`coefSVD`	logical indicating the need for SVD for the coeffient matrix in the output; used in ssvd estimation

Value

S3 `rrr` object, a list consisting of

`coef`	coefficient of rrr
`coef.ls`	coefficient of least square
`fitted`	fitted value of rrr
`fitted.ls`	fitted value of least square
`A`	right singular matrix
`Ad`	a vector of sigular values
`rank`	rank of the fitted rrr

Examples

```
Y <- matrix(rnorm(400), 100, 4)
X <- matrix(rnorm(800), 100, 8)
rfit <- rrr.fit(Y, X, nrank = 2)
coef(rfit)
```

rrr.leverage — *Leverage scores and Cook's distance in reduced-rank regression for model diagnostics*

Description

Compute leverage scores and Cook's distance for model diagnostics in `rrr` estimation.

Usage

```
rrr.leverage(Y, X = NULL, nrank = 1, qr.tol = 1e-07)
```

Arguments

`Y`	a matrix of response (n by q)
`X`	a matrix of covariate (n by p)
`nrank`	an integer specifying the desired rank
`qr.tol`	tolerence to be passed to 'qr'

Value

'rrr.leverage' returns a list containing a vector of leverages and a scalar of the degrees of freedom (sum of leverages). 'rrr.cooks' returns a list containing

`residuals`	resisuals matrix
`mse`	mean squared error
`leverage`	leverage
`cookD`	Cook's distance
`df`	degrees of freedom

References

Chen, K. Model diagnostics in reduced-rank estimation. *Statistics and Its interface*, 9, 469–484.

`rrr.sim1` *Simulation model 1*

Description

Similar to the the RSSVD simulation model in Chen, Chan, Stenseth (2012), JRSSB.

Usage

```
rrr.sim1(
  n = 50,
  p = 25,
  q = 25,
  nrank = 3,
  s2n = 1,
  sigma = NULL,
  rho_X = 0.5,
  rho_E = 0
)
```

Arguments

`n, p, q`	model dimensions
`nrank`	model rank
`s2n`	signal to noise ratio
`sigma`	error variance. If specfied, then s2n has no effect
`rho_X`	correlation parameter in the generation of predictors
`rho_E`	correlation parameter in the generation of random errors

Value

similated model and data

References

Chen, K., Chan, K.-S. and Stenseth, N. C. (2012) Reduced rank stochastic regression with a sparse singular value decomposition. *Journal of the Royal Statistical Society: Series B*, 74, 203–221.

rrr.sim2 *Simulation model 2*

Description

Similar to the the SRRR simulation model in Chen and Huang (2012), JASA

Usage

```
rrr.sim2(
  n = 100,
  p = 50,
  p0 = 10,
  q = 50,
  q0 = 10,
  nrank = 3,
  s2n = 1,
  sigma = NULL,
  rho_X = 0.5,
  rho_E = 0
)
```

Arguments

n	sample size
p	number of predictors
p0	number of relevant predictors
q	number of responses
q0	number of relevant responses
nrank	model rank
s2n	signal to noise ratio
sigma	error variance. If specfied, then s2n has no effect
rho_X	correlation parameter in the generation of predictors
rho_E	correlation parameter in the generation of random errors

Value

similated model and data

References

Chen, L. and Huang, J.Z. (2012) Sparse reduced-rank regression for simultaneous dimension reduction and variable selection. *Journal of the American Statistical Association*, 107:500, 1533–1545.

`rrr.sim3` *Simulation model 3*

Description

Generate data from a mixed-response reduced-rank regression model

Usage

```
rrr.sim3(
  n = 100,
  p = 30,
  q.mix = c(5, 20, 5),
  nrank = 2,
  intercept = rep(0.5, 30),
  mis.prop = 0.2
)
```

Arguments

`n`	sample size
`p`	number of predictors
`q.mix`	numbers of Gaussian, Bernolli and Poisson responses
`nrank`	model rank
`intercept`	a vector of intercept
`mis.prop`	missing proportion

Value

similated model and data

References

Chen, K., Luo, C., and Liang, J. (2017) Leveraging mixed and incomplete outcomes through a mixed-response reduced-rank regression. *Technical report*.

rrr.sim4 *Simulation model 4*

Description

Generate data from a mean-shifted reduced-rank regression model

Usage

```
rrr.sim4(
  n = 100,
  p = 12,
  q = 8,
  nrank = 3,
  s2n = 1,
  rho_X = 0,
  rho_E = 0,
  nout = 10,
  vout = NULL,
  voutsd = 2,
  nlev = 10,
  vlev = 10,
  vlevsd = NULL,
  SigmaX = "CorrCS",
  SigmaE = "CorrCS"
)
```

Arguments

n	sample size
p	number of predictors
q	numbers of responses
nrank	model rank
s2n	signal to noise ratio
rho_X	correlation parameter for predictors
rho_E	correlation parameter for errors
nout	number of outliers; should be smaller than n
vout	control mean-shifted value of outliers
voutsd	control mean-shifted magnitude of outliers
nlev	number of high-leverage outliers
vlev	control value of leverage
vlevsd	control magnitude of leverage
SigmaX	correlation structure of predictors
SigmaE	correlation structure of errors

Value

similated model and data

References

She, Y. and Chen, K. (2017) Robust reduced-rank regression. *Biometrika*, 104 (3), 633–647.

`rrr.sim5` *Simulation model 5*

Description

Generate data from a mean-shifted reduced-rank regression model

Usage

```
rrr.sim5(
  n = 40,
  p = 100,
  q = 50,
  nrank = 5,
  rx = 10,
  s2n = 1,
  rho_X = 0,
  rho_E = 0,
  nout = 10,
  vout = NULL,
  voutsd = 2,
  nlev = 10,
  vlev = 10,
  vlevsd = NULL,
  SigmaX = "CorrCS",
  SigmaE = "CorrCS"
)
```

Arguments

n	sample size
p	number of predictors
q	numbers of responses
nrank	model rank
rx	rank of the design matrix
s2n	signal to noise ratio
rho_X	correlation parameter for predictors
rho_E	correlation parameter for errors

nout	number of outliers; should be smaller than n
vout	control mean-shifted value of outliers
voutsd	control mean-shifted magnitude of outliers
nlev	number of high-leverage outliers
vlev	control value of leverage
vlevsd	control magnitude of leverage
SigmaX	correlation structure of predictors
SigmaE	correlation structure of errors

Value

similated model and data

References

She, Y. and Chen, K. (2017) Robust reduced-rank regression. *Biometrika*, 104 (3), 633–647.

---

rrs.fit — *Fitting reduced-rank ridge regression with given rank and shrinkage penalty*

Description

Fitting reduced-rank ridge regression with given rank and shrinkage penalty

Usage

```
rrs.fit(Y, X, nrank = min(ncol(Y), ncol(X)), lambda = 1, coefSVD = FALSE)
```

Arguments

Y	a matrix of response (n by q)
X	a matrix of covariate (n by p)
nrank	an integer specifying the desired rank
lambda	tunging parameter for the ridge penalty
coefSVD	logical indicating the need for SVD for the coeffient matrix int the output

Value

S3 `rrr` object, a list consisting of

`coef`	coefficient of rrs
`coef.ls`	coefficient of least square
`fitted`	fitted value of rrs
`fitted.ls`	fitted value of least square
`A`	right singular matrix
`Ad`	sigular value vector
`nrank`	rank of the fitted rrr

References

Mukherjee, A. and Zhu, J. (2011) Reduced rank ridge regression and its kernal extensions.

Mukherjee, A., Chen, K., Wang, N. and Zhu, J. (2015) On the degrees of freedom of reduced-rank estimators in multivariate regression. *Biometrika*, 102, 457–477.

Examples

```
library(rrpack)
Y <- matrix(rnorm(400), 100, 4)
X <- matrix(rnorm(800), 100, 8)
rfit <- rrs.fit(Y, X)
```

---

rssvd — *Reduced-rank regression with a sparse singular value decomposition*

Description

Reduced-rank regression with a sparse singular value decomposition using the iterative exclusive extraction algorithm.

Usage

```
rssvd(
  Y,
  X,
  nrank,
  ic.type = c("BIC", "BICP", "AIC"),
  orthX = FALSE,
  control = list(),
  screening = FALSE
)
```

Arguments

`Y`	response matrix
`X`	covariate matrix
`nrank`	integer specification of the desired rank
`ic.type`	character specifying which information criterion to use to select the best: 'BIC', 'BICP', and 'AIC'
`orthX`	logical indicating if X is orthogonal, in which case a faster algorithm is used
`control`	a list of parameters controlling the fitting process
`screening`	If TRUE, marginal screening via glm is performed before srrr fitting.

Details

The model fitting can be controled through argument `control`. The available elements include

- maxit: maximum number of iterations.
- epsilon: convergence tolerance.
- innerMaxit: maximum number of iterations for inner steps.
- innerEpsilon: convergence tolerance for inner steps.
- nlambda: number of tuning parameters.
- adaptive: if Ture, use adaptive penalization.
- gamma0: power parameter for constructing adaptive weights.
- minLambda: multiplicate factor to determine the minimum lambda.
- niter.eea: the number of iterations in the iterative exclusive extraction algorithm.
- df.tol: tolerance.

Value

S3 `rssvd.path` object, a list consisting of

`Upath`	solution path of U
`Vpath`	solution path of V
`Dpath`	solution path of D
`U`	estimated left singular matrix that is orthogonal
`V`	estimated right singular matrix that is orthogonal
`D`	estimated singular values such that C=UDVt
`rank`	estimated rank

References

Chen, K., Chan, K.-S. and Stenseth, N. C. (2012) Reduced rank stochastic regression with a sparse singular value decomposition. *Journal of the Royal Statistical Society: Series B*, 74, 203–221.

Examples

```
library(rrpack)
## Simulate data from a sparse factor regression model
p <- 50; q <- 50; n <- 100; nrank <- 3
mydata <- rrr.sim1(n, p, q, nrank, s2n = 1, sigma = NULL,
                   rho_X = 0.5, rho_E = 0.3)
fit1 <- with(mydata, rssvd(Y, X, nrank = nrank + 1))
summary(fit1)
plot(fit1)
```

---

sofar *Sparse orthogonal factor regression*

Description

Compute solution paths of sparse orthogonal factor regression

Usage

```
sofar(
  Y,
  X,
  nrank = 1,
  su = NULL,
  sv = NULL,
  ic.type = c("GIC", "BIC", "AIC", "GCV"),
  modstr = list(),
  control = list(),
  screening = FALSE
)
```

Arguments

Y	response matrix
X	covariate matrix
nrank	an integer specifying the desired rank/number of factors
su	a scaling vector for U such that $U^T U = diag(s_u)$.
sv	a scaling vector for V such that $V^T V = diag(s_v)$.
ic.type	select tuning method; the default is GIC
modstr	a list of internal model parameters controlling the model fitting
control	a list of internal computation parameters controlling optimization
screening	If TRUE, marginal screening via lasso is performed before sofar fitting.

Details

The model parameters can be specified through argument `modstr`. The available elements include

- mu: parameter in the augmented Lagrangian function.
- mugamma: increament of mu along iterations to speed up computation.
- WA: weight matrix for A.
- WB: weight matrix for B.
- Wd: weight matrix for d.
- wgamma: power parameter in constructing adaptive weights.

The model fitting can be controled through argument `control`. The avilable elements include

- nlam: number of lambda triplets to be used.
- lam.min.factor: set the smallest lambda triplets as a fraction of the estimation lambda.max triplets.
- lam.max.factor: set the largest lambda triplets as a multiple of the estimation lambda.max triplets.
- lam.AB.factor: set the relative penalty level between A/B and D.
- penA,penB,penD: if TRUE, penalty is applied.
- lamA: sequence of tuning parameters for A.
- lamB: sequence of tuning parameters for B.
- lamD: sequence of tuning parameters for d.
- methodA: penalty for penalizing A.
- methodB: penalty for penalizing B.
- epsilon: convergence tolerance.
- maxit: maximum number of iterations.
- innerEpsilon: convergence tolerance for inner subroutines.
- innerMaxit: maximum number of iterations for inner subroutines.
- sv.tol: tolerance for singular values.

Value

A `sofar` object containing

`call`	original function call
`Y`	input response matrix
`X`	input predictor matrix
`Upath`	solution path of U
`Dpath`	solution path of D
`Vpath`	solution path of D
`Rpath`	path of estimated rank
`icpath`	path of information criteria

lam.id	ids of selected lambda for GIC, BIC, AIC and GCV
p.index	ids of predictors which passed screening
q.index	ids of responses which passed screening
lamA	tuning sequence for A
lamB	tuning sequence for B
lamD	tuning sequence for D
U	estimated left singular matrix that is orthogonal (factor weights)
V	estimated right singular matrix that is orthogonal (factor loadings)
D	estimated singular values
rank	estimated rank

References

Uematsu, Y., Fan, Y., Chen, K., Lv, J., & Lin, W. (2019). SOFAR: large-scale association network learning. *IEEE Transactions on Information Theory*, 65(8), 4924–4939.

Examples

```
## Not run:
library(rrpack)
## Simulate data from a sparse factor regression model
p <- 100; q <- 50; n <- 100; nrank <- 3
mydata <- rrr.sim1(n, p, q, nrank, s2n = 1,
                   sigma = NULL, rho_X = 0.5, rho_E = 0.3)
Y <- mydata$Y
X <- mydata$X

fit1 <- sofar(Y, X, ic.type = "GIC", nrank = nrank + 2,
              control = list(methodA = "adlasso", methodB = "adlasso"))
summary(fit1)
plot(fit1)

fit1$U
crossprod(fit1$U) #check orthogonality
fit1$V
crossprod(fit1$V) #check orthogonality

## End(Not run)
```

---

srrr *Row-sparse reduced-eank regresssion*

Description

Row-sparse reduced-rank regresssion for a prespecified rank; produce a solution path for selecting predictors

Usage

```
srrr(
  Y,
  X,
  nrank = 2,
  method = c("glasso", "adglasso"),
  ic.type = c("BIC", "BICP", "AIC", "GCV", "GIC"),
  A0 = NULL,
  V0 = NULL,
  modstr = list(),
  control = list(),
  screening = FALSE
)
```

Arguments

`Y`	response matrix
`X`	covariate matrix
`nrank`	prespecified rank
`method`	group lasso or adaptive group lasso
`ic.type`	information criterion
`A0`	initial value
`V0`	initial value
`modstr`	a list of model parameters controlling the model fitting
`control`	a list of parameters for controlling the fitting process
`screening`	If TRUE, marginal screening via glm is performed before srrr fitting.

Details

Model parameters controlling the model fitting can be specified through argument `modstr`. The available elements include

- lamA: tuning parameter sequence.
- nlam: number of tuning parameters; no effect if `lamA` is specified.
- minLambda: minimum lambda value, no effect if `lamA` is specified.
- maxLambda: maxmum lambda value, no effect if lamA is specified.
- WA: adaptive weights. If NULL, the weights are constructed from RRR.
- wgamma: power parameter for constructing adaptive weights.

Similarly, the computational parameters controlling optimization can be specified through argument `control`. The available elements include

- epsilon: epsilonergence tolerance.
- maxit: maximum number of iterations.
- inner.eps: used in inner loop.
- inner.maxit: used in inner loop.

Value

A list of fitting results

References

Chen, L. and Huang, J. Z. (2012) Sparse reduced-rank regression for simultaneous dimension reduction and variable selection. *Journal of the American Statistical Association*. 107:500, 1533–1545.

Examples

```
library(rrpack)
p <- 100; n <- 100; nrank <- 3
mydata <- rrr.sim2(n, p, p0 = 10,q = 50, q0 = 10, nrank = 3,
                   s2n = 1, sigma = NULL, rho_X = 0.5, rho_E = 0)
fit1 <- with(mydata, srrr(Y, X, nrank = 3))
summary(fit1)
coef(fit1)
plot(fit1)
```

---

summary — *Summarize rrpack Objects*

Description

S3 methods summarizing objects generated by `rrpack`.

Usage

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

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

## S3 method for class 'r4'
summary(object, ...)

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

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

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

## S3 method for class 'cv.sofar'
```

```
summary(object, ...)

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

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

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

Arguments

`object`	Object generated from `rrpack`.
`...`	Other argumnts for future usage.

Index