Package ‘rrpack’

October 14, 2022

Title Reduced-Rank Regression

Version 0.1-13


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:

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>cv.mrrr</td>
<td>2</td>
</tr>
<tr>
<td>cv.mrr</td>
<td>4</td>
</tr>
<tr>
<td>cv.srrr</td>
<td>7</td>
</tr>
<tr>
<td>cv.sofar</td>
<td>5</td>
</tr>
<tr>
<td>mrrr</td>
<td>8</td>
</tr>
<tr>
<td>plot</td>
<td>10</td>
</tr>
<tr>
<td>r4</td>
<td>12</td>
</tr>
<tr>
<td>rrpcoefs</td>
<td>14</td>
</tr>
<tr>
<td>rrr</td>
<td>15</td>
</tr>
<tr>
<td>rrr.cookD</td>
<td>17</td>
</tr>
<tr>
<td>rrr.fit</td>
<td>18</td>
</tr>
<tr>
<td>rrr.leverage</td>
<td>19</td>
</tr>
<tr>
<td>rrr.sim1</td>
<td>20</td>
</tr>
<tr>
<td>rrr.sim2</td>
<td>21</td>
</tr>
<tr>
<td>rrr.sim3</td>
<td>22</td>
</tr>
<tr>
<td>rrr.sim4</td>
<td>23</td>
</tr>
<tr>
<td>rrr.sim5</td>
<td>24</td>
</tr>
<tr>
<td>rrs.fit</td>
<td>25</td>
</tr>
<tr>
<td>rssvd</td>
<td>26</td>
</tr>
<tr>
<td>sofard</td>
<td>28</td>
</tr>
<tr>
<td>srrr</td>
<td>30</td>
</tr>
<tr>
<td>summary</td>
<td>32</td>
</tr>
</tbody>
</table>

Index 34

---

**Description**

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

**Usage**

```r
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(),
```
cv.mrrr

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 controlling 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**

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

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>response matrix</td>
</tr>
<tr>
<td>X</td>
<td>covariate matrix</td>
</tr>
<tr>
<td>nfold</td>
<td>number of folds</td>
</tr>
<tr>
<td>maxrank</td>
<td>maximum rank allowed</td>
</tr>
<tr>
<td>norder</td>
<td>for constructing the folds</td>
</tr>
<tr>
<td>coefSVD</td>
<td>If TRUE, svd of the coefficient is retuned</td>
</tr>
</tbody>
</table>
Value

a list containing rr estimates from cross validation

References


Examples

```r
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

```r
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 $U^T U = \text{diag}(s_u)$
sv

a scaling vector for V such that \( V^T V = \text{diag}(s_v) \)

nfold

number of fold; used for cv.sofar

norder

observation orders to construct 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: increment 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 controlled through argument control. The available 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: maximum 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


---

**mrr**r

*Generalized or mixed-response reduced-rank regression*

**Description**

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

**Usage**

```r
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
ctrl1.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 controlling 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 mrr 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

```r
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)
```

Description

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

Usage

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

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

plot Scatter Plot
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 arguments for future usage.

Value

A 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`: solution path of regression coefficients
- `s.path`: solution path of sparse mean shifts
- `s.norm.path`: solution 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

Examples

```r
## 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

```r
## 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 arguments for future usage.

### Value

A numeric matrix.

---

### Description

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

### Usage

```r
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-constrained 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 degrees 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 tolerance of singular values; 'qr.tol' controls the tolerance 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 tolerance.
- qr.tol: QR decomposition tolerance.

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 matrix 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 vector of sum of squared errors
ic  a vector of information criterion
coeff  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


Examples

```r
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**

```r
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**

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

```r
trrr.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 coefficient 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 singular values
- **rank**: rank of the fitted rrr

Examples

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

Compute leverage scores and Cook’s distance for model diagnostics in *rrr* estimation.

**Usage**

```r
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`: tolerance 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`: residuals matrix
- `mse`: mean squared error
- `leverage`: leverage
- `cookD`: Cook’s distance
- `df`: degrees of freedom

**References**

Description

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

Usage

```r
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 specified, 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

Simulated model and data

References

Description

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

Usage

```r
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 specified, 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

simulated model and data

References

Description

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

Usage

```r
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

Simulated model and data

References

Simulation model 4

Description

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

Usage

```r
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
simulated model and data

References

---

**rrr.sim5**  
*Simulation model 5*

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

Usage

```r
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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>n</code></td>
<td>sample size</td>
</tr>
<tr>
<td><code>p</code></td>
<td>number of predictors</td>
</tr>
<tr>
<td><code>q</code></td>
<td>numbers of responses</td>
</tr>
<tr>
<td><code>nrank</code></td>
<td>model rank</td>
</tr>
<tr>
<td><code>rx</code></td>
<td>rank of the design matrix</td>
</tr>
<tr>
<td><code>s2n</code></td>
<td>signal to noise ratio</td>
</tr>
<tr>
<td><code>rho_X</code></td>
<td>correlation parameter for predictors</td>
</tr>
<tr>
<td><code>rho_E</code></td>
<td>correlation parameter for errors</td>
</tr>
</tbody>
</table>
### 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
simulated model and data

#### References

---

**rrs.fit**

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

#### Usage

```r
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**: tuning parameter for the ridge penalty
- **coefSVD**: logical indicating the need for SVD for the coefficient matrix in 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` singular value vector
- `nrank` rank of the fitted rrs

References


Examples

```r
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

```r
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 controlled 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 True, 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

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 $U^T U = \text{diag}(s_u)$.
sv  a scaling vector for V such that $V^T V = \text{diag}(s_v)$.
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`: increment 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 controlled through argument `control`. The available 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
Row-sparse reduced-rank regression 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: maximum 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.
summary

Value
A list of fitting results

References

Examples

```r
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**

Summary *rrpack* Objects

**Description**
S3 methods summarizing objects generated by *rrpack*.

**Usage**

```r
## 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**  

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>Object generated from rrpack.</td>
</tr>
<tr>
<td>...</td>
<td>Other arguments for future usage.</td>
</tr>
</tbody>
</table>
Index

coeff.cv.mrrr (rrpack-coef), 14
coeff.cv.rrr (rrpack-coef), 14
coeff.mrrr (rrpack-coef), 14
coeff.r4 (rrpack-coef), 14
coeff.rrr (rrpack-coef), 14
coeff.rssvd (rrpack-coef), 14
coeff.sofar (rrpack-coef), 14
coeff.srrr (rrpack-coef), 14
cv.mrrr, 2
cv.rrr, 4
cv.sofar, 5
cv.srrr, 7

mrrr, 8

plot, 10

r4, 12
rrpack-coef, 14
rrr, 15
rrr.cookD, 17
rrr.fit, 18
rrr.leverage, 19
rrr.sim1, 20
rrr.sim2, 21
rrr.sim3, 22
rrr.sim4, 23
rrr.sim5, 24
rrs.fit, 25
rssvd, 26

sofar, 28
srrr, 30
summary, 32