Package ‘tsvd’

March 24, 2015

Type Package

Title Thresholding-based SVD for multivariate reduced rank regression

Version 1.3

Date 2015-03-24

Author Xin Ma, Luo Xiao, and Wing H. Wong*

Maintainer Xin Ma <xm24@stanford.edu>

Description This package performs multivariate reduced rank regression with a sparse singular value decomposition on the coefficient matrix (T-SVD). Sparsity is achieved through iterative (hard) thresholding on the left and right singular vectors.

Depends MASS, glmnet

License GPL (>=2)

R topics documented:

<table>
<thead>
<tr>
<th>R topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>tsvd-package</td>
<td>2</td>
</tr>
<tr>
<td>MatrixMultiply</td>
<td>2</td>
</tr>
<tr>
<td>select_rank</td>
<td>3</td>
</tr>
<tr>
<td>soda</td>
<td>4</td>
</tr>
<tr>
<td>tsvd</td>
<td>5</td>
</tr>
<tr>
<td>tsvd_ini</td>
<td>7</td>
</tr>
<tr>
<td>tsvd_wrapper</td>
<td>9</td>
</tr>
</tbody>
</table>

Index 11
Description

Thresholding-based SVD for multivariate reduced rank regression. The development of microarray and next-generation sequencing technologies has enabled rapid quantification of various genome-wide features (genomic sequences, gene expressions, noncoding RNA expressions, methylation etc.) in a population of samples. Large consortiums have compiled genetic and molecular profiling data in an enormous number of tumors samples. There are some key characteristics in these regulatory relationships. First, multiple pathways are involved and can be viewed as independent programs of regulation. Second, combinatorial nature of the regulation in each program is likely to be sparse. Third, the number of potential predictors and responses often far exceed the sample size. To address these challenges, we develop this computationally efficient multivariate response regression model T-SVD.

Details

This package performs multivariate reduced rank regression with a sparse SVD on the coefficient matrix. The model is

\[ Y = XC + E, \]

where \( C \) is an unknown coefficient matrix. Let \( UDV^T \) be the singular value decomposition (SVD) of \( C \). The method in this package intends to obtain sparse estimates of \( U \) and \( V \).

Author(s)

Xin Ma, Luo Xiao, Wing H. Wong*

References


Description

This function gives the multiplication result of two matrices.

Usage

MatrixMultiply(A, B)

Arguments

\( A \) i by j matrix
\( B \) j by k matrix
select_rank

Value
The multiplication result

Author(s)
Xin Ma, Luo Xiao, Wing H. Wong*

References

Examples
Y <- matrix(rnorm(3), 1, 3)
X <- matrix(rnorm(15), 3, 5)
product <- MatrixMultiply(Y,X)

select_rank

Rank selection for multivariate reduced rank regression

Description
This function selects the rank of the coefficient matrix for multivariate response regression using the approach proposed by Bunea, She and Wegkamp (2011). Specifically, let

\[ P = X(X^TX)^{-}X^T, \]

where the generalized inverse is the Moore-Penrose pseudoinverse, and \( \lambda_k(Y^TPY) \) be the \( k \)-th eigenvalue of \( Y^TPY \). Then the selected rank is

\[ K = \max\{k : \lambda_k(Y^TPY) \geq c \max(q,n)\sigma^2\}, \]

where \( c \) is the constant and \( \sigma \) is the noise level of the data, estimated by \( 1.4826 \cdot \text{MAD}(\text{abs}(Y)) \).

Usage
select_rank(Y, X, sigma2 = NULL, constant = 4, abs.tol = 10^(-7), rel.tol = 1e-04)

Arguments
- \( Y \) n by q response matrix
- \( X \) n by p design matrix
- sigma2 random error variance in \( Y \)
- constant defaults to 4
- abs.tol absolute tolerance for numerical error, defaults to 1e-07
- rel.tol relative tolerance for numerical error, defaults to 1e-04

Value
The selected rank
Author(s)

Xin Ma <xm24@stanford.edu> Luo Xiao <lxiao@jhsph.edu> Wing H. Wong* <whwong@stanford.edu>

References


Examples

```r
n <- 100
p <- 200
q <- 100
rank <- 3
X <- matrix(rnorm(n*p),n,p)
U <- matrix(rnorm(p*rank),p,rank)
V <- matrix(rnorm(q*rank),q,rank)
D <- diag(rnorm(rank))
C <- U%*%D%*%t(V)
Y <- X%*%C + matrix(rnorm(n*q),n,q)
select_rank(Y,X)
```

soda

**Sparse Orthogonal Decomposition Algorithm**

**Description**

This function is similar to the QR decomposition of a numeric matrix but provides potentially more sparse singular vectors if the input matrix is sparse. If the input has no sparsity, then one gets the same result as the QR decomposition.

**Usage**

`soda(x)`

**Arguments**

- `x` a numeric matrix whose left singular vectors are to be computed

**Value**

- `Q` left singular vectors with zero rows removed
- `R` R matrix as in the QR decomposition
- `index` indices of the rows of `Q`
- `d` singular values of `x`
Author(s)
Xin Ma, Luo Xiao, Wing H. Wong*

References

See Also
qr

Examples

## Example 1: no sparsity in the data
## same as the QR decomposition
x <- matrix(rnorm(200),40,5)
Q <- qr.Q(qr(x))
Q1 <- soda(x)$Q
f <- function(x,y){ min(sum((x-y)*2),sum((x+y)*2))}
diff <- 0
for(r in 1:5) diff <- diff + f(Q[,r],Q1[,r])
print(diff)

## Example 2: sparse data
## different from the QR decomposition
x <- matrix(rnorm(900),30,3)
x[1:10,2] <- 0
x[1:20,3] <- 0
Q <- qr.Q(qr(x))
Q1 <- soda(x)$Q
print(cbind(x,Q,Q1),digits=3)

tsvd

Thresholding-based SVD for multivariate reduced rank regression

Description
This function performs (hard) thresholding-based SVD for multivariate reduced rank regression proposed by Ma, Xiao and Wong (2014).

Usage
tsvd(Y, X, rank, ini, BICtype = 2, thr = NULL, control = list(thr.lower = -10, thr.upper = -1, thr.by = 0.5))
Arguments

\( Y \)  
\( n \) by \( q \) response matrix

\( X \)  
\( n \) by \( p \) design matrix

\( \text{rank} \)  
rank of the coefficient matrix

\( \text{ini} \)  
initial estimate of the coefficient matrix

\( \text{BICtype} \)  
two BIC types

\( \text{thr} \)  
a pair of tuning parameters or a 2-column matrix of tuning parameters to search over

\( \text{control} \)  
a list of arguments for defining the sequence of tuning parameters; see details

Details

There are two types of BICs that can be used. BICtype 1:

\[
\log(\text{SSE}) + \log(q \times n)/(q \times n) \text{df},
\]

where \( \text{df} \) is the total number of non-zero entries in \( U, V \) and \( D \), subtracted by squared rank. BIC type 2:

\[
\log(\text{SSE}) + \log(q \times n)/(q \times n) \text{df}_v + \log(r \times n)/(r \times n) \text{df}_u,
\]

where \( r \) is the rank, \( \text{df}_v \) is the number of non-zero entries in \( V \), subtracted by \( r \times (r - 1)/2 \), and \( \text{df}_u \) is the number of non-zero entries in \( U \), subtracted by \( r \times (r - 1)/2 \). The second BIC type performs better for variable selection on \( U \).

Value

\( \text{est} \)  
estimate of the coefficient matrix in “SparseSVD” structure

\( \text{C} \)  
estimate of the coefficient matrix

\( \text{thr} \)  
selected tuning parameters

\( \text{BICtype} \)  
BICtype used

\( \text{BIC} \)  
value of BIC

Author(s)

Xin Ma, Luo Xiao, Wing H. Wong*

References


Examples

\[
n <- 100
\]
\[
p <- 150
\]
\[
q <- 150
\]

```r
## generate a sparse coefficient matrix
lnorm <- function(a, l=1) (sum(abs(a)^2))^(1/2)  # normaling function
U <- matrix(ncol= 3,nrow= p);
V <- matrix(ncol= 3,nrow= q);
```
\[ V[,1] \leftarrow c(sample(c(-1,1),5,replace= TRUE),rep(0,20)); \]
\[ V[,2] \leftarrow c(rep(0,12),sample(c(-1,1),8,replace= TRUE),rep(0,5)); \]
\[ V[,3] \leftarrow c(rep(0,6),V[1:2,1],-V[3:4,1],sample(c(-1,1),2,replace= TRUE), \]
\[ -V[15:16,2],V[13:14,2],rep(0,9)) \]
\[ V[,1] \leftarrow V[,1]/lnorm(V[,1],2); \]
\[ V[,2] \leftarrow V[,2]/lnorm(V[,2],2); \]
\[ V[,3] \leftarrow V[,3]/lnorm(V[,3],2); \]
\[ U[,1] \leftarrow c(sample(c(1,-1),5,replace= TRUE)*runif(5,0.7,1),rep(0,20)); \]
\[ U[,2] \leftarrow c(rep(0,5),sample(c(1,-1),5,replace= TRUE)*runif(5,0.7,1),rep(0,15)); \]
\[ U[,3] \leftarrow c(rep(0,10),sample(c(1,-1),5,replace= TRUE)*runif(5,0.7,1),rep(0,10)); \]
\[ U[,1] \leftarrow U[,1]/lnorm(U[,1],2); \]
\[ U[,2] \leftarrow U[,2]/lnorm(U[,2],2); \]
\[ U[,3] \leftarrow U[,3]/lnorm(U[,3],2); \]
\[ D \leftarrow diag(c(20,10,5)); \]
\[ C \leftarrow U%*%D%*%t(V); \]
## generate data
\[ X \leftarrow matrix(rnorm(n*p),n,p) \]
\[ E \leftarrow matrix(rnorm(n*q),n,q) \]
\[ sigma2 \leftarrow sum(diag(t(C)%*%C))/(n*q) \] ## so that the signal to noise ratio is 1
\[ Y \leftarrow X%*%C+ sqrt(sigma2)*E \]
## obtain estimate
\[ ini1 \leftarrow tsvd_ini(Y,X,rank=3) \]
\[ ans1 \leftarrow tsvd(Y,X,rank=3,ini=ini1,BICtype=2) \]
\[ ini2 \leftarrow tsvd_ini(Y,X,rank=3,ini=ans1$est,method=2) \]
\[ ans2 \leftarrow tsvd(Y,X,rank=3,ini=ini2,BICtype=2) \]
\[ print(sum((ans2$C-C)^2)) \]

---

**tsvd_ini**

An initial estimator for multivariate reduced rank regression

**Description**

This function provides an intial estimate for multivariate reduced rank regression.

**Usage**

```r
tsvd_ini(Y, X, rank, ini = NULL, method = 1)
```

**Arguments**

- **Y**
  - n by q response matrix
- **X**
  - n by p design matrix
- **rank**
  - rank of the estimate
- **ini**
  - an initial estimate of the coefficient matrix in “SparseSVD” structure
- **method**
  - method = 1: rank-truncated ridge regression estimate; method = 2: assuming V is known from ini, estimate U by lasso(implemented by glmnet) for which the tuning parameter is selected by BIC.
Value

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>rank</td>
<td>rank</td>
</tr>
<tr>
<td>u</td>
<td>left singular vectors with zero rows removed</td>
</tr>
<tr>
<td>v</td>
<td>right singular vectors with zero rows removed</td>
</tr>
<tr>
<td>d</td>
<td>singular values</td>
</tr>
<tr>
<td>u.index</td>
<td>indices of rows of u</td>
</tr>
<tr>
<td>v.index</td>
<td>indices of rows of v</td>
</tr>
</tbody>
</table>

Author(s)

Xin Ma, Luo Xiao, Wing H. Wong*

References


Examples

```r
n <- 100
p <- 150
q <- 150

## generate a sparse coefficient matrix
lnorm <- function(a, l=1) (sum(abs(a)^2))^(1/2) #normalizing function
U <- matrix(ncol= 3, nrow= p);
V <- matrix(ncol= 3, nrow= q);
V[,1] <- c(sample(c(-1,1),5,replace= TRUE),rep(0,20));
V[,2] <- c(rep(0,12),sample(c(-1,1),8,replace= TRUE),rep(0 ,5));
V[,3] <- c(rep(0,6),V[1:2,1],-V[3:4,1],sample(c(-1,1),2,replace= TRUE),
-V[15:16,2],V[13:14,2],rep(0,9))
V[,1] <- V[,1]/lnorm(V[,1],2);
V[,2] <- V[,2]/lnorm(V[,2],2);
V[,3] <- V[,3]/lnorm(V[,3],2);
U[,1] <- c(sample(c(1,-1),5,replace= TRUE)*runif(5,0.7,1),rep(0,20));
U[,2] <- c(rep(0,5),sample(c(-1,1),5,replace= TRUE)*runif(5,0.7,1),rep(0,15));
U[,3] <- c(rep(0,10),sample(c(1,-1),5,replace= TRUE)*runif(5,0.7,1),rep(0,10));
U[,1] <- U[,1]/lnorm(U[,1],2);
U[,2] <- U[,2]/lnorm(U[,2],2);
U[,3] <- U[,3]/lnorm(U[,3],2);
D <- diag(c(20,10,5));

C <- UX*%DX*%t(V);  
## generate data
X <- matrix(rnorm(n*p),n,p)
E <- matrix(rnorm(n*q),n,q)
sigma2 <- sum(diag(t(C)%*%X))/(n*q)  ## so that the signal to noise ratio is 1
Y <- X*%C+ sqrt(sigma2)*E

## obtain initial estimate
```
The model is

\[ Y = XC + E, \]

where \( C \) is an unknown coefficient matrix. Let \( UDV^T \) be the singular value decomposition (SVD) of \( C \). The method in this package intends to obtain sparse estimates of \( U \) and \( V \).

If \( \text{rank} \) is \( \text{NULL} \), an estimate is obtained from \( \text{select_rank} \).

The wrapper has four steps: (1) obtain an initial estimate from rank-truncated ridge regression; (2) run \( \text{tsvd} \); (3) with \( V \) from step (2), obtain another estimate of \( U \) and \( D \) from lasso; (4) run \( \text{tsvd} \).

There are two types of BICs that can be used. **BICtype 1:**

\[ \log(\text{SSE}) + \log(q \times n)/(q \times n)df, \]

where \( df \) is the total number of non-zero entries in \( U, V \) and \( D \), subtracted by squared rank. **BIC type 2:**

\[ \log(\text{SSE}) + \log(q \times n)/(q \times n)df_v + \log(r \times n)/(r \times n)df_u, \]

where \( r \) is the rank, \( df_v \) is the number of non-zero entries in \( V \), subtracted by \( r \times (r - 1)/2 \), and \( df_u \) is the number of non-zero entries in \( U \), subtracted by \( r \times (r - 1)/2 \). The second BIC type performs better for variable selection on \( U \).
Value

- `est`: estimate of the coefficient matrix in “SparseSVD” structure
- `C`: estimate of the coefficient matrix
- `thr`: selected tuning parameters
- `BICtype`: BICtype used
- `BIC`: value of BIC

Author(s)

Xin Ma, Luo Xiao, Wing H. Wong*

References


Examples

```r
n <- 100  
p <- 150  
q <- 150  

## generate a sparse coefficient matrix
lnorm <- function(a, l=1) (sum(abs(a)^2))^(1/2) #normaling function
U <- matrix(ncol=3, nrow= p);  
V <- matrix(ncol=3, nrow= q);  
V[,1] <- c(sample(c(-1,1),5,replace= TRUE),rep(0,20));  
V[,2] <- c(rep(0,12),sample(c(-1,1),8,replace= TRUE),rep(0,5));  
V[,3] <- c(rep(0,6),V[,2,1],-V[,3,1],sample(c(-1,1),2,replace= TRUE),  
         -V[15:16,2],V[,13:14,2],rep(0,9));
V[,1] <- V[,1]/lnorm(V[,1],2);  
V[,2] <- V[,2]/lnorm(V[,2],2);  
V[,3] <- V[,3]/lnorm(V[,3],2);

U[,1] <- c(sample(c(1,-1),5,replace= TRUE)*runif(5,.7,1),rep(0,20));  
U[,2] <- c(rep(0,5),sample(c(1,-1),5,replace= TRUE)*runif(5,.7,1),rep(0,15));  
U[,3] <- c(rep(0,10),sample(c(1,-1),5,replace= TRUE)*runif(5,.7,1),rep(0,10));  
U[,1] <- U[,1]/lnorm(U[,1],2);  
U[,2] <- U[,2]/lnorm(U[,2],2);  
U[,3] <- U[,3]/lnorm(U[,3],2);  
D <- diag(c(5,1));

C <- U%*%D%*%t(V);  
## generate data
X <- matrix(rnorm(n*p),n,p)  
E <- matrix(rnorm(n*q),n,q)
sigma2 <- sum(diag(t(C)%*%C))/(n*q) # so that the signal to noise ratio is 1
Y <- X%*%C + sqrt(sigma2)*E

## obtain estimate
ans <- tsvd_wrapper(Y,X,BICtype=2) ##rank is estimated
print(sum((ans$C-C)^2))
```
Index

- Topic **array**
  - soda, 4

- Topic **multiplication**
  - MatrixMultiply, 2

- Topic **multivariate**
  - select_rank, 3
  - tsvd, 5
  - tsvd_ini, 7
  - tsvd_wrapper, 9

- Topic **package**
  - tsvd-package, 2

- Topic **regression**
  - MatrixMultiply, 2
  - select_rank, 3, 9
  - tsvd, 5
  - tsvd_ini, 7
  - tsvd_wrapper, 9

MatrixMultiply, 2

qr, 5

select_rank, 3, 9
soda, 4

tsvd, 5
tsvd-package, 2
tsvd_ini, 7
tsvd_wrapper, 9