Package ‘SPCAvRP’

May 4, 2019

Type Package

Title Sparse Principal Component Analysis via Random Projections (SPCAvRP)

Version 0.4

Date 2019-05-01

Author Milana Gataric, Tengyao Wang and Richard J. Samworth

Maintainer Milana Gataric <m.gataric@statslab.cam.ac.uk>


Depends R (>= 3.0.0), parallel, MASS

License GPL-3


NeedsCompilation no

Repository CRAN

RoxygenNote 6.1.1

Date/Publication 2019-05-03 23:00:04 UTC

R topics documented:

SPCAvRP .......................................................... 2
SPCAvRP_deflation ........................................... 4
SPCAvRP_subspace ........................................... 6

Index 9
SPCAvRP

Computes the leading eigenvector using the SPCAvRP algorithm

Description

Computes 1-sparse leading eigenvector of the sample covariance matrix, using $A \times B$ random axis-aligned projections of dimension $d$. For the multiple component estimation use \texttt{SPCAvRP\_subspace} or \texttt{SPCAvRP\_deflation}.

Usage

\begin{verbatim}
SPCAvRP(data, cov = FALSE, l = 20, A = 600, B = 200,
center_data = TRUE, parallel = FALSE,
cluster_type = "PSOCK", cores = 1, machine_names = NULL)
\end{verbatim}

Arguments

\begin{itemize}
  \item \texttt{data} Either the data matrix ($p \times n$) or the sample covariance matrix ($p \times p$).
  \item \texttt{cov} TRUE if data is given as a sample covariance matrix.
  \item \texttt{l} Desired sparsity level in the final estimator (see Details).
  \item \texttt{d} The dimension of the random projections (see Details).
  \item \texttt{A} Number of projections over which to aggregate (see Details).
  \item \texttt{B} Number of projections in a group from which to select (see Details).
  \item \texttt{center_data} TRUE if the data matrix should be centered (see Details).
  \item \texttt{parallel} TRUE if the selection step should be computed in parallel by uses package "parallel".
  \item \texttt{cluster_type} If parallel == TRUE, this can be "PSOCK" or "FORK" (cf. package "parallel").
  \item \texttt{cores} If parallel == TRUE and cluster_type == "FORK", number of cores to use.
  \item \texttt{machine_names} If parallel == TRUE, the names of the computers on the network.
\end{itemize}

Details

This function implements the SPCAvRP algorithm for the principal component estimation (Algorithm 1 in the reference given below).

If the true sparsity level $k$ is known, use $l = k$ and $d = k$.

If the true sparsity level $k$ is unknown, $l$ can take an array of different values and then the estimators of the corresponding sparsity levels are computed. The final choice of $l$ can then be done by the user via inspecting the explained variance computed in the output value or via inspecting the output \texttt{importance\_scores}. The default choice for $d$ is 20, but we suggest choosing $d$ equal to or slightly larger than 1.

It is desirable to choose $A$ (and $B = \text{ceiling}(A/3)$) as big as possible subject to the computational budget. In general, we suggest using $A = 300$ and $B = 100$ when the dimension of data is a few hundreds, while $A = 600$ and $B = 200$ when the dimension is on order of 1000.
If `center_data == TRUE` and data is given as a data matrix, the first step is to center it by executing `scale(data, center_data, FALSE)`, which subtracts the column means of data from their corresponding columns.

If `parallel == TRUE`, the parallelised SPCAvRP algorithm is used. We recommend to use this option if \( p, A \) and \( B \) are very large.

**Value**

Returns a list of three elements:

- **vector**: A matrix of dimension \( p \times \text{length}(l) \) with columns as the estimated eigenvectors of sparsity level \( l \).
- **value**: An array with \( \text{length}(l) \) eigenvalues corresponding to the estimated eigenvectors returned in vector.
- **importance_scores**: An array of length \( p \) with importance scores for each variable 1 to \( p \).

**Author(s)**

Milana Gataric, Tengyao Wang and Richard J. Samworth

**References**


**Examples**

```r
p <- 100  # data dimension
k <- 10   # true sparsity level
n <- 1000 # number of observations
v1 <- c(rep(1/sqrt(k), k), rep(0, p-k)) # true principal component
Sigma <- 2*tcrossprod(v1) + diag(p)   # population covariance
mu <- rep(0, p)                        # population mean
loss = function(u,v){
  # the loss function
  sqrt(abs(1-sum(v*u)^2))
}
set.seed(1)
X <- mvrnorm(n, mu, Sigma) # data matrix
spcavrp <- SPCAvRP(data = X, cov = FALSE, l = k, d = k, A = 200, B = 70)
spcavrp.loss <- loss(v1,spcavrp$vector)
print(paste0("estimation loss when l=d=k=10, A=200, B=70: ", spcavrp.loss))

# choosing sparsity level l if k unknown:
#spcavrp.choose1 <- SPCAvRP(data = X, cov = FALSE, l = c(1:30), d = 15, A = 200, B = 70)
#plot(1:p,spcavrp.choose1$importance_scores,xlab='variable',ylab='w',
#     main='choosing 1 when k unknown: \n importance scores w')
#plot(1:30,spcavrp.choose1$value,xlab='l',ylab='Var_l',
#     main='choosing 1 when k unknown: \n explained variance Var_l')
```
SPCAvRP_deflation

**Description**

Computes \( m \) leading eigenvectors of the sample covariance matrix which are sparse and orthogonal, using the modified deflation scheme in conjunction with the SPCAvRP algorithm.

**Usage**

```r
SPCAvRP_deflation(data, cov = FALSE, m, l = 20, A = 600, B = 200, center_data = TRUE)
```

**Arguments**

- `data`: Either the data matrix \((p \times n)\) or the sample covariance matrix \((p \times p)\).
- `cov`: TRUE if data is given as a sample covariance matrix.
- `m`: The number of principal components to estimate.
- `l`: The array of length \( m \) with the desired sparsity of \( m \) principle components (see Details).
- `d`: The dimension of the random projections (see Details).
- `A`: Number of projections over which to aggregate (see Details).
- `B`: Number of projections in a group from which to select (see Details).
- `center_data`: TRUE if the data matrix should be centered (see Details).

**Details**

This function implements the modified deflation scheme in conjunction with SPCAvRP (Algorithm 2 in the reference given below).

If the true sparsity level is known and for each component is equal to \( k \), use \( d = k \) and \( l = \text{rep}(k, m) \). Sparsity levels of different components may take different values. If \( k \) is unknown, appropriate \( k \) could be chosen from an array of different values by inspecting the explained variance for one component at the time and by using SPCAvRP in a combination with the deflation scheme implemented in SPCAvRP_deflation.

It is desirable to choose \( A \) (and \( B = \text{ceiling}(A/3) \)) as big as possible subject to the computational budget. In general, we suggest using \( A = 300 \) and \( B = 100 \) when the dimension of data is a few hundreds, while \( A = 600 \) and \( B = 200 \) when the dimension is on order of 1000.

If `center_data == TRUE` and data is given as a data matrix, the first step is to center it by executing `scale(data, center_data, FALSE)`, which subtracts the column means of data from their corresponding columns.
SPCAvRP_deflation

Value

Returns a list of two elements:

- **vector**: A matrix whose \( m \) columns are the estimated eigenvectors.
- **value**: An array with \( m \) estimated eigenvalues.

Author(s)

Milana Gataric, Tengyao Wang and Richard J. Samworth

References


See Also

SPCAvRP, SPCAvRP_subspace

Examples

\[
p <- 50 \# data dimension \\
k <- 8 \# true sparsity of each component \\
v1 <- 1/sqrt(k)*c(rep(1, k), rep(0, p-k)) \# first principal component (PC) \\
v2 <- 1/sqrt(k)*c(rep(0,4), 1, -1, 1, -1, rep(1,4), rep(0,p-12)) \# 2nd PC \\
v3 <- 1/sqrt(k)*c(rep(0,6), 1, -rep(1,4), rep(1,3), rep(0, p-14)) \# 3rd PC \\
Sigma <- diag(p) + 40*tcrossprod(v1) + 20*tcrossprod(v2) + 5*tcrossprod(v3) \# population covariance \\
mu <- rep(0, p) \# population mean \\
n <- 2000 \# number of observations \\
loss = function(u,v){ \\
  sqrt(abs(sum(v*u)^R)) \\
} \\
loss_sub = function(U,V){ \\
  U <- qr.Q(qr(U)); V <- qr.Q(qr(V)); \\
  norm(tcrossprod(U)-tcrossprod(V),"2") \\
} \\
set.seed(1) \\
X <- mvrnorm(n, mu, Sigma) \# data matrix \\
spcavrp.def <- SPCAvRP_deflation(data = X, cov = FALSE, m = 2, l = rep(k,2), \\
  d = k, A = 200, B = 70, center_data = FALSE) \\
subspace_estimation<-data.frame(
  loss_sub(matrix(c(v1,v2),ncol=2),spcavrp.def$vector), 
  loss(spcavrp.def$vector[,1],v1), 
  loss(spcavrp.def$vector[,2],v2), 
  crossprod(spcavrp.def$vector[,1],spcavrp.def$vector[,2]))
rownames(subspace_estimation)<-c("loss_sub","loss_v1","loss_v2","inner_prod")
print(subspace_estimation)<-c("")
print(subspace_estimation)
SPCAvRP_subspace  Computes the leading eigenspace using the SPCAvRP algorithm for the eigenspace estimation

Description

Computes \( m \) leading eigenvectors of the sample covariance matrix which are sparse and orthogonal, using \( A \times B \) random axis-aligned projections of dimension \( d \).

Usage

```r
SPCAvRP_subspace(data, cov = FALSE, m, l, d = 20,
A = 600, B = 200, center_data = TRUE)
```

Arguments

- `data` Either the data matrix \((p \times n)\) or the sample covariance matrix \((p \times p)\).
- `cov` TRUE if data is given as a sample covariance matrix.
- `m` The dimension of the eigenspace, i.e. the number of principal components to compute.
- `l` Desired sparsity level of the eigenspace (i.e. the number of non-zero rows in output$vector) (see Details).
- `d` The dimension of the random projections (see Details).
- `A` Number of projections over which to aggregate (see Details).
- `B` Number of projections in a group from which to select (see Details).
- `center_data` TRUE if the data matrix should be centered (see Details).

Details

This function implements the SPCAvRP algorithm for the eigenspace estimation (Algorithm 3 in the reference given below).

If the true sparsity level \( k \) of the eigenspace is known, use \( l = k \) and \( d = k \).

If the true sparsity level \( k \) of the eigenspace is unknown, the appropriate choice of \( l \) can be done, for example, by running the algorithm (for any \( l \)) and inspecting the computed output importance_scores. The default choice for \( d \) is 20, but we suggest choosing \( d \) equal to or slightly larger than \( l \).

It is desirable to choose \( A \) (and \( B = \text{ceiling}(A/3) \)) as big as possible subject to the computational budget. In general, we suggest using \( A = 300 \) and \( B = 100 \) when the dimension of data is a few hundreds, while \( A = 600 \) and \( B = 200 \) when the dimension is on order of 1000.

If `center_data == TRUE` and data is given as a data matrix, the first step is to center it by executing `scale(data, center_data, FALSE)`, which subtracts the column means of data from their corresponding columns.
SPCAvRP_subspace

Value

Returns a list of two elements:

- **vector**: A matrix whose \( m \) columns are the estimated eigenvectors.
- **value**: An array with \( m \) estimated eigenvalues.
- **importance_scores**: An array of length \( p \) with importance scores for each variable 1 to \( p \).

Author(s)

Milana Gataric, Tengyao Wang and Richard J. Samworth

References


See Also

SPCAvRP, SPCAvRP_deflation

Examples

```r
p <- 50 # data dimension
k1 <- 8 # sparsity of each individual component
v1 <- 1/sqrt(k1)*c(rep(1, k1), rep(0, p-k1)) # first principal component (PC)
v2 <- 1/sqrt(k1)*c(rep(0, 4), 1, -1, 1, -1, rep(1, 4), rep(0, p-12)) # 2nd PC
v3 <- 1/sqrt(k1)*c(rep(0, 6), 1, -rep(1, 4), rep(1, 3), rep(0, p-14)) # 3rd PC
Sigma <- diag(p) + 40*tcrossprod(v1) + 20*tcrossprod(v2) + 5*tcrossprod(v3) # population covariance
mu <- rep(0, p) # population mean
n <- 2000 # number of observations
loss = function(u,v){
  sqrt(abs(1-sum(v*u)^2))
}
loss_sub = function(U,V){
  U <- qr.Q(qr(U)); V <- qr.Q(qr(V))
  norm(tcrossprod(U)-tcrossprod(V),"2")
}
set.seed(1)
X <- mvrnorm(n, mu, Sigma) # data matrix

spcavrp_sub <- SPCAvRP_subspace(data = X, cov = FALSE, m = 2, l = 12, d = 12,
                                A = 200, B = 70, center_data = FALSE)

subspace_estimation<-data.frame(
  loss_sub(matrix(c(v1,v2),ncol=2),spcavrp_sub$vector),
  loss(spcavrp_sub$vector[,1],v1),
  loss(spcavrp_sub$vector[,2],v2),
  crossprod(spcavrp_sub$vector[,1],spcavrp_sub$vector[,2]))
colnames(subspace_estimation)<-c("loss_sub","loss_v1","loss_v2","inner_prod")
```
rownames(subspace_estimation)<-c"
"
print(subspace_estimation)

plot(1:p,spcavrp.sub$importance_scores,xlab='variable',ylab='w',
    main='importance scores w \n (may use to choose l when k unknown)')
Index

SPCAvRP, 2, 5, 7
SPCAvRP_deflation, 2, 4, 7
SPCAvRP_subspace, 2, 5, 6