Package ‘SPCAvRP’

July 26, 2018

Type Package

Title Sparse Principal Component Analysis via Random Projections (SPCAvRP)

Version 0.3

Date 2018-07-26

Author Milana Gataric, Tengyao Wang and Richard J. Samworth

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

Description Implements the SPCAvRP algorithm, developed and analysed in ``Sparse principal component analysis via random projections'' Gataric, M., Wang, T. and Samworth, R. J. (2018) <arXiv:1712.05630>. The algorithm is based on the aggregation of eigenvector information from carefully-selected random projections of the sample covariance matrix.

Depends R (>= 3.0.0), parallel, MASS

License GPL-3

URL https://arxiv.org/abs/1712.05630

NeedsCompilation no

Repository CRAN

RoxygenNote 6.0.1

Date/Publication 2018-07-26 13:40:03 UTC

R topics documented:

`final_estimator` *Computes the leading eigenvector from its support*

Description

Computes the leading eigenvector of the sample covariance matrix given the indices of variables ranked by their importance and desired sparsity level.

Usage

```
final_estimator(data, cov, l, ranking)
```

Arguments

`data`	Either the data matrix or the sample covariance matrix.
`cov`	`TRUE` if data is given as a sample covariance matrix.
`l`	Desired sparsity of the final estimator (see Details).
`ranking`	Original variables ranked by their importance.

Details

If true sparsity level `k` is known use `l = k`. If `k` is unknown, `l` can be an array of different values and then the eigenvectors of the corrsponding sparsity levels are returned.

Value

Returns a list of two elements:

`vector`	A vector or a matrix with `length(l)` columns as the estimated eigenvectors of sparsity level `l`.
`value`	An array with `length(l)` estimated eigenvalues.

Author(s)

Milana Gataric, Tengyao Wang and Richard J. Samworth

References

Milana Gataric, Tengyao Wang and Richard J. Samworth (2018) Sparse principal component analysis via random projections https://arxiv.org/abs/1712.05630

Examples

```
p <- 80
k <- 8
n <- 1000
v1 <- c(rep(1/sqrt(k), k), rep(0,p-k))
Sigma <- 2*tcrossprod(v1) + diag(p)
mu <- rep(0, p)
X <- mvrnorm(n, mu, Sigma)
Sigma_hat <- 1/n*crossprod(X)

A <- 200
B <- 100
d <- 10

rand_ind <- matrix(replicate(A*B,sample.int(p,d)), nrow = A*B, byrow = TRUE)
cov_projections <- project_covariance(data = Sigma_hat, cov = TRUE, rand_ind)
ranking <- SPCAvRP_ranking(cov_projections, rand_ind, p, A)

output <- final_estimator(data = Sigma_hat, cov = TRUE, l = (5:11), ranking)
df <- data.frame(5:11,output$value); colnames(df) <- c('l','eigenvalue')
print(df)
```

`project_covariance` *Projects the sample covariance*

Description

Projects the sample covariance matrix along given axis-aligned projections.

Usage

```
project_covariance(data, cov, rand_ind)
```

Arguments

`data`	Either the data matrix or the sample covariance matrix (see Details).
`cov`	`TRUE` if data is given as a sample covariance matrix.
`rand_ind`	Matrix whose rows are the indices of non-zero entries of axis-aligned projections.

Details

If the dimension of data is very large, it might be faster if the data matrix is provided as the input.

Value

Returns a list of the sample covariance projections:

`projections[[1]]`
: projected sample covariance along indices of `rand_ind[1,]`

...

`projections[[nrow(rand_ind)]]`
: projected sample covariance along indices of `rand_ind[nrow(rand_ind),]`

Author(s)

Milana Gataric, Tengyao Wang and Richard J. Samworth

References

Milana Gataric, Tengyao Wang and Richard J. Samworth (2018) Sparse principal component analysis via random projections https://arxiv.org/abs/1712.05630

Examples

```
p <-  50 # dimension of data
k <- 5 # sparsity level
n <- 1000 # number of observations
v1 <- c(rep(1/sqrt(k), k), rep(0,p-k))
Sigma_hat <- 1/n*crossprod(mvrnorm(n, rep(0,p), tcrossprod(v1)+diag(p)))

N <- 1000 # number of projections
d <- k # dimension of projections
rand_ind <- matrix(replicate(N,sample.int(p,d)), nrow = N, byrow = TRUE) # axis-aligned projections

cov_projections <- project_covariance(data = Sigma_hat, cov = TRUE, rand_ind)
```

`select_projection` *Selects the best projection*

Description

Selects the projection yielding the largest eigenvalue among one group of B different d-dimensional axis-aligned projections generated uniformly at random.

Usage

```
select_projection(data, cov = TRUE, p, d, B)
```

Arguments

data	Either the data matrix or the sample covariance matrix.
cov	FALSE if data is given as a data matrix.
p	Original dimension of the data.
d	Dimension of the random projections.
B	Number of random projections to generate and select from.

Details

If p is very large, it might be faster if the data matrix is provided as the input.

Value

Returns eigenvector `v_hat_star` corresponding to the projected covariance yielding the largest eigenvalue among B different d-dimensional axis-aligned random projections.

Author(s)

Milana Gataric, Tengyao Wang and Richard J. Samworth

References

Milana Gataric, Tengyao Wang and Richard J. Samworth (2018) Sparse principal component analysis via random projections https://arxiv.org/abs/1712.05630

Examples

```
p <-  100
k <- 10
n <- 1000
v1 <- c(rep(1/sqrt(k), k), rep(0,p-k))
Sigma <- 2*tcrossprod(v1) + diag(p)
mu <- rep(0, p)

X <- mvrnorm(n, mu, Sigma)

v_hat_star <- select_projection(data = 1/n*crossprod(X), cov = TRUE, p, d = k, B = 100)
```

---

select_projections_subspace

Selects the best projections for the subspace estimation

Description

Selects A projections yielding the largest r-th eigenvalue among B different random projections, for `r=1:s`.

Usage

```
select_projections_subspace(data, rand_ind, s, p, A)
```

Arguments

`data`	A list of projected covariances to select from (in the form of the output of `project_covariance`).
`rand_ind`	Corresponding projections used to generate `data`.
`s`	The number of eigenvalues to estimate.
`p`	The original dimension of samples.
`A`	The number of projections to select.

Value

Returns matrix `v_hat_stars` with `A` columns that correspond to `s` eigenvectors yielding the largest eigenvalues.

Author(s)

Milana Gataric, Tengyao Wang and Richard J. Samworth

References

Milana Gataric, Tengyao Wang and Richard J. Samworth (2018) Sparse principal component analysis via random projections `https://arxiv.org/abs/1712.05630`

`SPCAvRP` *Computes the leading eigenvector using the SPCAvRP algorithm*

Description

Computes `l`-sparse leading eigenvector of the sample covariance matrix, using `A` groups of `B` random axis-aligned projections of dimension `d`.

Usage

```
SPCAvRP(data, cov = FALSE, l, d = 10, A = 300, B = 100,
center_data = TRUE)
```

Arguments

`data`	Either the data matrix or the sample covariance matrix.
`cov`	`TRUE` if data is given as a sample covariance matrix.
`l`	Desired sparsity level in the final estimator (see Details).
`d`	The dimension of the random projections.
`A`	Number of projections over which to aggregate.
`B`	Number of projections in a group from which to select.
`center_data`	`TRUE` if the data matrix should be centered.

Details

This function implements the SPCAvRP algorithm.

If the true sparsity level `k` is known, use `d = k` and `l = k`. If `k` is unknown, the default choice for `d` is `10`, while `l` can take an array of different values and then the estimators of the corresponding sparsity levels are computed.

It is desirable to choose `A` 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.

Value

Returns a list of two elements:

`vector`	A vector or a matrix with `length(l)` columns as the estimated eigenvectors of sparsity level `l`.
`value`	An array with `length(l)` estimated eigenvalues.

Author(s)

Milana Gataric, Tengyao Wang and Richard J. Samworth

References

Milana Gataric, Tengyao Wang and Richard J. Samworth (2018) Sparse principal component analysis via random projections `https://arxiv.org/abs/1712.05630`

See Also

`SPCAvRP_parallel`, `SPCAvRP_ranking`

Examples

```
p <-  100 # dimension of data
k <- 10 # true sparsity level
n <- 1000 # number of observations
v1 <- c(rep(1/sqrt(k), k), rep(0,p-k)) # leading eigenvector
Sigma <- 2*tcrossprod(v1) + diag(p) # population covariance
mu <- rep(0, p) # population mean
X <- mvrnorm(n, mu, Sigma) # data matrix

spca <- SPCAvRP(data = X, cov = FALSE, l = k, d = k, A = 200, B = 70, center_data = FALSE)
spca$vector
spca$value
```

SPCAvRP_deflation *Computes the leading eigenvectors using the modified deflation scheme*

Description

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

Usage

```
SPCAvRP_deflation(data, cov = FALSE, s, l, d = 10,
A = 300, B = 100, center_data = TRUE)
```

Arguments

data	Either the data matrix or the sample covariance matrix.
cov	TRUE if data is given as a sample covariance matrix.
s	The number of eigenvectors to compute.
l	The array of lenght s with the desired sparsity levels in the final estimators.
d	The dimension of the random projections.
A	Number of projections over which to aggregate.
B	Number of projections in a group from which to select.
center_data	TRUE if the data matrix should be centered.

Details

This function implements the modified deflation scheme in conjunction with the SPCAvRP in order to compute s sparse eigenvectors that are orthogonal. If possible, use SPCAvRP_subspace instead.

If the true sparsity level is known and for each component is equal to k, use d = k and l = rep(k,s). 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` 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.

Value

Returns a list of two elements:

`vector`	A matrix whose `s` columns are the estimated eigenvectors.
`value`	An array with `s` estimated eigenvalues.

Author(s)

Milana Gataric, Tengyao Wang and Richard J. Samworth

References

Milana Gataric, Tengyao Wang and Richard J. Samworth (2018) Sparse principal component analysis via random projections `https://arxiv.org/abs/1712.05630`

See Also

`SPCAvRP`, `SPCAvRP_subspace`

Examples

```
p <- 50
k <- 8
theta <- 40
v1 <- c(rep(1/sqrt(k), k), rep(0, p-k))
theta2 <- 20
v2 <- c(rep(0,4), 1/sqrt(k), -1/sqrt(k), 1/sqrt(k), -1/sqrt(k), rep(1/sqrt(k),4), rep(0,p-12))
theta3 <- 5
v3 <- c(rep(0,6), 1/sqrt(k), -rep(1/sqrt(k),4), rep(1/sqrt(k),3), rep(0,p-14))
Sigma <- diag(p) + theta*tcrossprod(v1) + (theta2)*tcrossprod(v2) + (theta3)*tcrossprod(v3)
mu <- rep(0, p)
n <- 2000
X <- mvrnorm(n, mu, Sigma)

spcarp <- SPCAvRP_deflation(data = X, cov = FALSE, s = 1, l = k, d = k,
                            A = 300, B = 100, center_data = FALSE)
```

SPCAvRP_parallel *Parallel implementation of the SPCAvRP algorithm*

Description

Computes `l`-sparse leading eigenvector of the sample covariance matrix, by parallel selection of `A` projections, where each projection is selected from a group of `B` random projections of dimension `d`.

Usage

```
SPCAvRP_parallel(data, cov = FALSE, l, d = 10, A = 300, B = 100,
center_data = TRUE, cluster_type = "PSOCK", cores = 1, machine_names = NULL)
```

Arguments

`data`	Either the data matrix or the sample covariance matrix.
`cov`	`TRUE` if data is given as a sample covariance matrix.
`l`	Desired sparsity level in the final estimator (see Details).
`d`	The dimension of the random projections.
`A`	Number of projections over which to aggregate.
`B`	Number of projections in a group from which to select.
`center_data`	`TRUE` if the data matrix should be centered.
`cluster_type`	Can be `"PSOCK"` or `"FORK"` (cf. package `"parallel"`).
`cores`	Number of cores to use if `clustertype=="FORK"`.
`machine_names`	Names of computers on the network if `clustertype=="PSOCK"`.

Details

This function implements the parallelised SPCAvRP algorithm, by calling `'select_projection'` `A` times in parallel. We recommend to use this function if `p`, `A` and `B` are large; otherwise use `SPCAvRP`.

If the true sparsity level `k` is known, use `d = k` and `l = k`. If `k` is unknown, the default choice for `d` is `10`, while `l` can take an array of different values and then the estimators of the corresponding sparsity levels are computed.

It is desirable to choose `A` 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.

Value

Returns a list of two elements:

`vector`	A vector or a matrix with `length(l)` columns as the estimated eigenvectors of sparsity level `l`.
`value`	An array with `length(l)` estimated eigenvalues.

Author(s)

Milana Gataric, Tengyao Wang and Richard J. Samworth

References

Milana Gataric, Tengyao Wang and Richard J. Samworth (2018) Sparse principal component analysis via random projections `https://arxiv.org/abs/1712.05630`

See Also

`SPCAvRP`

Examples

```
p <-  100 # dimension of data
k <- 10 # true sparsity level
n <- 1000 # number of observations
v1 <- c(rep(1/sqrt(k), k), rep(0,p-k)) # leading eigenvector
Sigma <- 2*tcrossprod(v1) + diag(p) # population covariance
mu <- rep(0, p) # population mean
X <- mvrnorm(n, mu, Sigma) # data matrix

spca <- SPCAvRP_parallel(data = X, cov = FALSE, l = k, d = k, A = 200, B = 70,
                          center_data = FALSE, cluster_type = "PSOCK")
```

`SPCAvRP_ranking` *Ranks the variables*

Description

Ranks the original variables according to their importance in maximising the explained variance in the data by aggregating over selected projections of the sample covariance matrix.

Usage

```
SPCAvRP_ranking(data, rand_ind, p, A)
```

Arguments

`data`	A list of projected covariances (see Details).
`rand_ind`	Corresponding axis-aligned projections (see Details).
`p`	The dimension of the data.
`A`	Number of projections over which to aggregate.

Details

This function divides given projections into `A` groups and selects the best one from each group. This is then followed by an aggregation step. Data is given as a list of d-dimensional projections of p-dimensional sample covariance matrix, as generated by the function `project_covariance`. Corresponding axis-aligned projections are given as a matrix whose rows are the indices of the projection's non-zero entries.

Value

Returns the vector of indices corresponding to variables ranked by their importance in maximising the explained variance.

Author(s)

Milana Gataric, Tengyao Wang and Richard J. Samworth

References

Milana Gataric, Tengyao Wang and Richard J. Samworth (2018) Sparse principal component analysis via random projections `https://arxiv.org/abs/1712.05630`

See Also

`SPCAvRP`

Examples

```
p <-  50 # dimension of data
k <- 5 # true sparsity level
n <- 1000 # number of observations
v1 <- c(rep(1/sqrt(k), k), rep(0,p-k)) # leading eigenvector
Sigma <- tcrossprod(v1) + diag(p) # population covariance
mu <- rep(0, p) # population mean
X <- mvrnorm(n, mu, Sigma) # data matrix
Sigma_hat <- 1/n*crossprod(X) # sample covariance matrix

A <- 200 # number of projections over which to aggregate
B <- 100 # number of projections in a group from which to select
d <- k # dimension of projections
rand_ind <- matrix(replicate(A*B,sample.int(p,d)), nrow = A*B, byrow = TRUE) # random projections

cov_projections <- project_covariance(data = Sigma_hat, cov = TRUE, rand_ind)
```

```
ranking <- SPCAvRP_ranking(cov_projections, rand_ind, p, A)
print(ranking)
```

`SPCAvRP_subspace` *Computes the leading eigenspace using the SPCAvRP algorithm for eigenspace estimation*

Description

Computes `s` leading eigenvectors of the sample covariance matrix which are sparse and orthogonal, using `A` groups of `B` random axis-aligned projections of dimension `d`.

Usage

```
SPCAvRP_subspace(data, cov = FALSE, s, l, d = 10,
A = 300, B = 100, center_data = TRUE)
```

Arguments

`data`	Either the data matrix or the sample covariance matrix.
`cov`	`TRUE` if data is given as a sample covariance matrix.
`s`	The dimension of the eigenspace, i.e the number of principal components.
`l`	The array of length `s` with the desired sparsity levels in the final estimators.
`d`	The dimension of the random projections.
`A`	Number of projections over which to aggregate.
`B`	Number of projections in a group from which to select.
`center_data`	`TRUE` if the data matrix should be centered.

Details

This function implements the SPCAvRP algorithm for eigenspace estimation.

If the true sparsity level is known and for each component is equal to `k`, use `d = k` and `l = rep(k,s)`. 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 `SPCAvRP_deflation`.

It is desirable to choose `A` 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.

Value

Returns a list of two elements:

`vector`	A matrix whose s columns are the estimated eigenvectors.
`value`	An array with s estimated eigenvalues.

Author(s)

Milana Gataric, Tengyao Wang and Richard J. Samworth

References

Milana Gataric, Tengyao Wang and Richard J. Samworth (2018) Sparse principal component analysis via random projections https://arxiv.org/abs/1712.05630

See Also

SPCAvRP, SPCAvRP_deflation

Examples

```
p <- 50
k <- 8
theta <- 40
v1 <- c(rep(1/sqrt(k), k), rep(0, p-k))
theta2 <- 20
v2 <- c(rep(0,4), 1/sqrt(k), -1/sqrt(k), 1/sqrt(k), -1/sqrt(k), rep(1/sqrt(k),4), rep(0,p-12))
theta3 <- 5
v3 <- c(rep(0,6), 1/sqrt(k), -rep(1/sqrt(k),4), rep(1/sqrt(k),3), rep(0,p-14))
Sigma <- diag(p) + theta*tcrossprod(v1) + (theta2)*tcrossprod(v2) + (theta3)*tcrossprod(v3)
mu <- rep(0, p)
n <- 2000
X <- mvrnorm(n, mu, Sigma)

loss = function(u,v){
  sqrt(abs(1-sum(v*u)^2))
}
loss_sub = function(U,V){
  V<-qr.Q(qr(V))
  norm(tcrossprod(U)-tcrossprod(V),"2")
}

s <- 2
spcarp <- SPCAvRP_subspace(data = X, cov = FALSE, s, l = rep(k,s), d = k,
                           A = 200, B = 100, center_data = FALSE)

subspace_estimation<-data.frame(
  loss(spcarp$vector[,1],v1),
  loss(spcarp$vector[,2],v2),
  loss_sub(matrix(c(v1,v2),ncol=s),spcarp$vector),
  crossprod(spcarp$vector[,1],spcarp$vector[,2]))
```

```
colnames(subspace_estimation)<-c("loss_1","loss_2","loss_sub","inp")
rownames(subspace_estimation)<-c("")
subspace_estimation
```

Index