# Package ‘sparsepca’

April 11, 2018

**Type** Package

**Title** Sparse Principal Component Analysis (SPCA)

**Version** 0.1.2

**Author** N. Benjamin Erichson, Peng Zheng, and Sasha Aravkin

**Maintainer** N. Benjamin Erichson <erichson@uw.edu>

**Description** Sparse principal component analysis (SPCA) attempts to find sparse weight vectors (loadings), i.e., a weight vector with only a few 'active' (nonzero) values. This approach provides better interpretability for the principal components in high-dimensional data settings. This is, because the principal components are formed as a linear combination of only a few of the original variables. This package provides efficient routines to compute SPCA. Specifically, a variable projection solver is used to compute the sparse solution. In addition, a fast randomized accelerated SPCA routine and a robust SPCA routine is provided. Robust SPCA allows to capture grossly corrupted entries in the data. The methods are discussed in detail by N. Benjamin Erichson et al. (2018) <arXiv:1804.00341>.

**License** GPL (>= 3)

**Encoding** UTF-8

**LazyData** true

**URL** https://github.com/erichson/spca

**BugReports** https://github.com/erichson/spca/issues

**Imports** rsvd

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2018-04-11 08:17:42 UTC

## R topics documented:

robspca . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2
rspca . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
spca . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7

**Index** **10**

robspca *Robust Sparse Principal Component Analysis (robspca).*

## Description

Implementation of robust SPCA, using variable projection as an optimization strategy.

## Usage

```
robspca(X, k = NULL, alpha = 1e-04, beta = 1e-04, gamma = 100,
  center = TRUE, scale = FALSE, max_iter = 1000, tol = 1e-05,
  verbose = TRUE)
```

## Arguments

| | |
|---|---|
| `X` | array_like;<br>a real $(n, p)$ input matrix (or data frame) to be decomposed. |
| `k` | integer;<br>specifies the target rank, i.e., the number of components to be computed. |
| `alpha` | float;<br>Sparsity controlling parameter. Higher values lead to sparser components. |
| `beta` | float;<br>Amount of ridge shrinkage to apply in order to improve conditioning. |
| `gamma` | float;<br>Sparsity controlling parameter for the error matrix S. Smaller values lead to a larger amount of noise removeal. |
| `center` | bool;<br>logical value which indicates whether the variables should be shifted to be zero centered (TRUE by default). |
| `scale` | bool;<br>logical value which indicates whether the variables should be scaled to have unit variance (FALSE by default). |
| `max_iter` | integer;<br>maximum number of iterations to perform before exiting. |
| `tol` | float;<br>stopping tolerance for the convergence criterion. |
| `verbose` | bool;<br>logical value which indicates whether progress is printed. |

## Details

Sparse principal component analysis is a modern variant of PCA. Specifically, SPCA attempts to find sparse weight vectors (loadings), i.e., a weight vector with only a few 'active' (nonzero) values. This approach leads to an improved interpretability of the model, because the principal components are formed as a linear combination of only a few of the original variables. Further, SPCA avoids overfitting in a high-dimensional data setting where the number of variables $p$ is greater than the number of observations $n$.

Such a parsimonious model is obtained by introducing prior information like sparsity promoting regularizers. More concreatly, given an $(n, p)$ data matrix $X$, robust SPCA attemps to minimize the following objective function:

$$f(A, B) = \frac{1}{2}\|X - XBA^\top - S\|^2_F + \psi(B) + \gamma\|S\|_1$$

where $B$ is the sparse weight matrix (loadings) and $A$ is an orthonormal matrix. $\psi$ denotes a sparsity inducing regularizer such as the LASSO ($\ell_1$ norm) or the elastic net (a combination of the $\ell_1$ and $\ell_2$ norm). The matrix $S$ captures grossly corrupted outliers in the data.

The principal components $Z$ are formed as

$$Z = XB$$

and the data can be approximately rotated back as

$$\tilde{X} = ZA^\top$$

The print and summary method can be used to present the results in a nice format.

## Value

`spca` returns a list containing the following three components:

| | |
|---|---|
| `loadings` | array_like;<br>sparse loadings (weight) vector; $(p, k)$ dimensional array. |
| `transform` | array_like;<br>the approximated inverse transform; $(p, k)$ dimensional array. |
| `scores` | array_like;<br>the principal component scores; $(n, k)$ dimensional array. |
| `sparse` | array_like;<br>sparse matrix capturing outliers in the data; $(n, p)$ dimensional array. |
| `eigenvalues` | array_like;<br>the approximated eigenvalues; $(k)$ dimensional array. |
| `center, scale` | array_like;<br>the centering and scaling used. |

## Author(s)

N. Benjamin Erichson, Peng Zheng, and Sasha Aravkin

## References

1 N. B. Erichson, P. Zheng, K. Manohar, S. Brunton, J. N. Kutz, A. Y. Aravkin. "Sparse Principal Component Analysis via Variable Projection." Submitted to IEEE Journal of Selected Topics on Signal Processing (2018). (available at ‘arXiv https://arxiv.org/abs/1804.00341).

## See Also

rspca, spca

## Examples

```
# Create artifical data
m <- 10000
V1 <- rnorm(m, 0, 290)
V2 <- rnorm(m, 0, 300)
V3 <- -0.1*V1 + 0.1*V2 + rnorm(m,0,100)

X <- cbind(V1,V1,V1,V1, V2,V2,V2,V2, V3,V3)
X <- X + matrix(rnorm(length(X),0,1), ncol = ncol(X), nrow = nrow(X))

# Compute SPCA
out <- robspca(X, k=3, alpha=1e-3, beta=1e-5, gamma=5, center = TRUE, scale = FALSE, verbose=0)
print(out)
summary(out)
```

---

rspca *Randomized Sparse Principal Component Analysis (rspca).*

---

## Description

Randomized accelerated implementation of SPCA, using variable projection as an optimization strategy.

## Usage

```
rspca(X, k = NULL, alpha = 1e-04, beta = 1e-04, center = TRUE,
  scale = FALSE, max_iter = 1000, tol = 1e-05, o = 20, q = 2,
  verbose = TRUE)
```

## Arguments

| | |
|---|---|
| X | array_like;<br>a real $(n, p)$ input matrix (or data frame) to be decomposed. |
| k | integer;<br>specifies the target rank, i.e., the number of components to be computed. |
| alpha | float;<br>Sparsity controlling parameter. Higher values lead to sparser components. |
| beta | float;<br>Amount of ridge shrinkage to apply in order to improve conditioning. |
| center | bool;<br>logical value which indicates whether the variables should be shifted to be zero centered (TRUE by default). |

`scale` bool;
logical value which indicates whether the variables should be scaled to have unit variance (FALSE by default).

`max_iter` integer;
maximum number of iterations to perform before exiting.

`tol` float;
stopping tolerance for the convergence criterion.

`o` integer;
oversampling parameter (default $o = 20$).

`q` integer;
number of additional power iterations (default $q = 2$).

`verbose` bool;
logical value which indicates whether progress is printed.

## Details

Sparse principal component analysis is a modern variant of PCA. Specifically, SPCA attempts to find sparse weight vectors (loadings), i.e., a weight vector with only a few 'active' (nonzero) values. This approach leads to an improved interpretability of the model, because the principal components are formed as a linear combination of only a few of the original variables. Further, SPCA avoids overfitting in a high-dimensional data setting where the number of variables $p$ is greater than the number of observations $n$.

Such a parsimonious model is obtained by introducing prior information like sparsity promoting regularizers. More concreatly, given an $(n, p)$ data matrix $X$, SPCA attemps to minimize the following objective function:

$$f(A, B) = \frac{1}{2}\|X - XBA^\top\|_F^2 + \psi(B)$$

where $B$ is the sparse weight (loadings) matrix and $A$ is an orthonormal matrix. $\psi$ denotes a sparsity inducing regularizer such as the LASSO ($\ell_1$ norm) or the elastic net (a combination of the $\ell_1$ and $\ell_2$ norm). The principal components $Z$ are formed as

$$Z = XB$$

and the data can be approximately rotated back as

$$\tilde{X} = ZA^\top$$

The print and summary method can be used to present the results in a nice format.

## Value

`spca` returns a list containing the following three components:

`loadings` array_like;
sparse loadings (weight) vector; $(p, k)$ dimensional array.

| | |
|---|---|
| `transform` | array_like;<br>the approximated inverse transform; $(p, k)$ dimensional array. |
| `scores` | array_like;<br>the principal component scores; $(n, k)$ dimensional array. |
| `eigenvalues` | array_like;<br>the approximated eigenvalues; $(k)$ dimensional array. |
| `center, scale` | array_like;<br>the centering and scaling used. |

## Note

This implementation uses randomized methods for linear algebra to speedup the computations. $o$ is an oversampling parameter to improve the approximation. A value of at least 10 is recommended, and $o = 20$ is set by default.

The parameter $q$ specifies the number of power (subspace) iterations to reduce the approximation error. The power scheme is recommended, if the singular values decay slowly. In practice, 2 or 3 iterations achieve good results, however, computing power iterations increases the computational costs. The power scheme is set to $q = 2$ by default.

If $k > (min(n, p)/4)$, a the deterministic `spca` algorithm might be faster.

## Author(s)

N. Benjamin Erichson, Peng Zheng, and Sasha Aravkin

## References

1 N. B. Erichson, P. Zheng, K. Manohar, S. Brunton, J. N. Kutz, A. Y. Aravkin. "Sparse Principal Component Analysis via Variable Projection." Submitted to IEEE Journal of Selected Topics on Signal Processing (2018). (available at ‘arXiv https://arxiv.org/abs/1804.00341).

1 N. B. Erichson, S. Voronin, S. Brunton, J. N. Kutz. "Randomized matrix decompositions using R." Submitted to Journal of Statistical Software (2016). (available at ‘arXiv http://arxiv.org/abs/1608.02148).

## See Also

`spca`, `robspca`

## Examples

```
# Create artifical data
m <- 10000
V1 <- rnorm(m, 0, 290)
V2 <- rnorm(m, 0, 300)
V3 <- -0.1*V1 + 0.1*V2 + rnorm(m,0,100)

X <- cbind(V1,V1,V1,V1, V2,V2,V2,V2, V3,V3)
X <- X + matrix(rnorm(length(X),0,1), ncol = ncol(X), nrow = nrow(X))
```

```
# Compute SPCA
out <- rspca(X, k=3, alpha=1e-3, beta=1e-3, center = TRUE, scale = FALSE, verbose=0)
print(out)
summary(out)
```

---

spca *Sparse Principal Component Analysis (spca).*

---

## Description

Implementation of SPCA, using variable projection as an optimization strategy.

## Usage

```
spca(X, k = NULL, alpha = 1e-04, beta = 1e-04, center = TRUE,
  scale = FALSE, max_iter = 1000, tol = 1e-05, verbose = TRUE)
```

## Arguments

| | |
|---|---|
| X | array_like;<br>a real $(n, p)$ input matrix (or data frame) to be decomposed. |
| k | integer;<br>specifies the target rank, i.e., the number of components to be computed. |
| alpha | float;<br>Sparsity controlling parameter. Higher values lead to sparser components. |
| beta | float;<br>Amount of ridge shrinkage to apply in order to improve conditioning. |
| center | bool;<br>logical value which indicates whether the variables should be shifted to be zero centered (TRUE by default). |
| scale | bool;<br>logical value which indicates whether the variables should be scaled to have unit variance (FALSE by default). |
| max_iter | integer;<br>maximum number of iterations to perform before exiting. |
| tol | float;<br>stopping tolerance for the convergence criterion. |
| verbose | bool;<br>logical value which indicates whether progress is printed. |

## Details

Sparse principal component analysis is a modern variant of PCA. Specifically, SPCA attempts to find sparse weight vectors (loadings), i.e., a weight vector with only a few 'active' (nonzero) values. This approach leads to an improved interpretability of the model, because the principal components are formed as a linear combination of only a few of the original variables. Further, SPCA avoids overfitting in a high-dimensional data setting where the number of variables $p$ is greater than the number of observations $n$.

Such a parsimonious model is obtained by introducing prior information like sparsity promoting regularizers. More concreatly, given an $(n, p)$ data matrix $X$, SPCA attemps to minimize the following objective function:

$$f(A, B) = \frac{1}{2}\|X - XBA^\top\|_F^2 + \psi(B)$$

where $B$ is the sparse weight (loadings) matrix and $A$ is an orthonormal matrix. $\psi$ denotes a sparsity inducing regularizer such as the LASSO ($\ell_1$ norm) or the elastic net (a combination of the $\ell_1$ and $\ell_2$ norm). The principal components $Z$ are formed as

$$Z = XB$$

and the data can be approximately rotated back as

$$\tilde{X} = ZA^\top$$

The print and summary method can be used to present the results in a nice format.

## Value

`spca` returns a list containing the following three components:

| | |
|---|---|
| `loadings` | array_like;<br>sparse loadings (weight) vector; $(p, k)$ dimensional array. |
| `transform` | array_like;<br>the approximated inverse transform; $(p, k)$ dimensional array. |
| `scores` | array_like;<br>the principal component scores; $(n, k)$ dimensional array. |
| `eigenvalues` | array_like;<br>the approximated eigenvalues; $(k)$ dimensional array. |
| `center, scale` | array_like;<br>the centering and scaling used. |

## Author(s)

N. Benjamin Erichson, Peng Zheng, and Sasha Aravkin

## References

1 N. B. Erichson, P. Zheng, K. Manohar, S. Brunton, J. N. Kutz, A. Y. Aravkin. "Sparse Principal Component Analysis via Variable Projection." Submitted to IEEE Journal of Selected Topics on Signal Processing (2018). (available at ‘arXiv https://arxiv.org/abs/1804.00341).

## See Also

rspca, robspca

## Examples

```
# Create artifical data
m <- 10000
V1 <- rnorm(m, 0, 290)
V2 <- rnorm(m, 0, 300)
V3 <- -0.1*V1 + 0.1*V2 + rnorm(m,0,100)

X <- cbind(V1,V1,V1,V1, V2,V2,V2,V2, V3,V3)
X <- X + matrix(rnorm(length(X),0,1), ncol = ncol(X), nrow = nrow(X))

# Compute SPCA
out <- spca(X, k=3, alpha=1e-3, beta=1e-3, center = TRUE, scale = FALSE, verbose=0)
print(out)
summary(out)
```

# Index


robspca, *2*, *6*, *9*
rspca, *4*, 4, *9*

spca, *4*, *6*, 7