# Package 'msPCA'

June 25, 2026

**Type** Package

**Title** Sparse Principal Component Analysis with Multiple Principal Components

**Version** 0.5.0

**Date** 2026-06-15

**Description** Implements an algorithm for computing multiple sparse principal components of a dataset. The method is based on Cory-Wright and Pauphilet ``Sparse PCA with Multiple Principal Components" (2026) <doi:10.1287/opre.2023.0598>. The algorithm uses an iterative deflation heuristic with a truncated power method applied at each iteration to compute sparse principal components with controlled sparsity.

**License** MIT + file LICENSE

**URL** https://jeanpauphilet.github.io/msPCA/

**Imports** Rcpp (>= 1.0.11)

**Suggests** datasets, knitr, rmarkdown

**LinkingTo** Rcpp, RcppEigen

**Encoding** UTF-8

**VignetteBuilder** knitr

**Config/roxygen2/version** 8.0.0

**NeedsCompilation** yes

**Author** Ryan Cory-Wright [aut, cph] (ORCID: <https://orcid.org/0000-0002-4485-0619>), Jean Pauphilet [aut, cre, cph] (ORCID: <https://orcid.org/0000-0001-6352-0984>)

**Maintainer** Jean Pauphilet <jpauphilet@london.edu>

**Repository** CRAN

**Date/Publication** 2026-06-25 16:30:02 UTC

# Contents



---

`feasibility_violation_off` *Feasibility Violation*

---

## Description

Computes the feasibility violation defined as $\sum_{t>s} |u_t^\top u_s|$ if orthogonality constraints are enforced (`feasibilityConstraintType = 0`) and $\sum_{t>s} |u_t^\top C u_s|$ if zero-correlation constraints are enforced (`feasibilityConstraintType = 1`).

## Usage

```
feasibility_violation_off(C, U, feasibilityConstraintType)
```

## Arguments

| | |
|---|---|
| `C` | A matrix. The correlation or covariance matrix (p x p). |
| `U` | A matrix. Each column corresponds to a p-dimensional PC. |
| `feasibilityConstraintType` | An integer. Type of feasibility constraints to be enforced. 0: orthogonality constraints; 1: uncorrelatedness constraints. |

## Value

A float.

## Examples

```
TestMat <- cor(mtcars)
mspcares <- mspca(TestMat, r = 2, ks = c(4, 4), verbose = FALSE)
feasibility_violation_off(TestMat, mspcares$x_best, 0)
```

## fraction_variance_explained

*Fraction of Variance Explained*

### Description

Computes the fraction of variance explained (variance explained normalized by the trace of the covariance/correlation matrix) by a set of PCs.

### Usage

```
fraction_variance_explained(C, U)
```

### Arguments

| | |
|---|---|
| C | A matrix. The correlation or covariance matrix (p x p). |
| U | A matrix. The matrix containing the r PCs (p x r). |

### Value

A float.

### Examples

```
TestMat <- cor(mtcars)
mspcares <- mspca(TestMat, r = 2, ks = c(4, 4), verbose = FALSE)
fraction_variance_explained(TestMat, mspcares$x_best)
```

## fraction_variance_explained_perPC

*Fraction of Variance Explained Per PC*

### Description

Computes the fraction of variance explained (variance explained normalized by the trace of the covariance/correlation matrix) by each PC.

### Usage

```
fraction_variance_explained_perPC(C, U)
```

### Arguments

| | |
|---|---|
| C | A matrix. The correlation or covariance matrix (p x p). |
| U | A matrix. The matrix containing the r PCs (p x r). |

## Value

A numeric vector of length `ncol(U)`.

---

`mspca` *Multiple Sparse PCA*

---

## Description

Returns multiple sparse principal components of a dataset using an iterative deflation heuristic. As in the `elasticnet` package, the data is passed as a single argument `M` whose interpretation is set by `type`: `"Sigma"` (the default) treats `M` as a covariance/correlation matrix (p x p) and `"X"` treats `M` as a raw data matrix (n observations x p variables). With `type = "X"` the algorithm operates on the data directly via the products $X^\top(X\beta)$ and never forms the p x p matrix, which is substantially more scalable when $n \ll p$.

## Usage

```
mspca(
  M,
  r,
  ks,
  type = c("Sigma", "X"),
  feasibilityConstraintType = 0,
  verbose = TRUE,
  maxIter = 200,
  feasibilityTolerance = 1e-04,
  stallingTolerance = 1e-08,
  timeLimitTPM = 20,
  maxRestartTPM = 30,
  minRestartTPM = 20,
  center = TRUE,
  scale = TRUE,
  divisor = c("n-1", "n"),
  checkPSD = TRUE,
  symTolerance = 1e-08,
  psdTolerance = 1e-08
)
```

## Arguments

| | |
|---|---|
| `M` | A matrix. The data, interpreted according to `type`: a covariance/ correlation matrix (p x p) when `type = "Sigma"`, or a raw data matrix (n x p) when `type = "X"`. |
| `r` | An integer. Number of principal components (PCs) to be computed. |
| `ks` | An integer vector. Target sparsity of each PC. |

`type` (optional) Either "Sigma" (default; `M` is a covariance/correlation matrix) or "X" (`M` is a raw data matrix).

`feasibilityConstraintType`
(optional) An integer. Type of feasibility constraints to be enforced. 0: orthogonality constraints; 1: uncorrelatedness constraints. Default 0.

`verbose` (optional) A Boolean. Controls console output. Default TRUE.

`maxIter` (optional) An integer. Maximum number of iterations of the algorithm. Default 200.

`feasibilityTolerance`
(optional) A float. Tolerance for constraint violation (orthogonality/uncorrelatedness, according to `feasibilityConstraintType`). Default 1e-4.

`stallingTolerance`
(optional) A float. Controls the objective improvement below which the algorithm is considered to have stalled. Default 1e-8.

`timeLimitTPM` (optional) An integer. Maximum time in seconds for the truncated power method (inner iteration). Default 20.

`maxRestartTPM` (optional) An integer. Number of random restarts of the truncated power method (inner iteration) for the first outer iteration. Default 30.

`minRestartTPM` (optional) An integer. Number of random restarts of the truncated power method (inner iteration) for outer iterations >= 2. Default 20.

`center` (optional, type = "X") A Boolean. Center the columns of `M` before computing the covariance. Default TRUE.

`scale` (optional, type = "X") A Boolean. Scale the columns of `M` to unit variance, i.e. operate on the correlation matrix. Default TRUE.

`divisor` (optional, type = "X") Either "n-1" (default, sample covariance, matches `cov`/`cor`) or "n" (population covariance). Default "n-1".

`checkPSD` (optional, type = "Sigma") A Boolean. Verify that `M` is positive semidefinite. Default TRUE.

`symTolerance` (optional, type = "Sigma") A float. Tolerance for the symmetry check on `M`. Default 1e-8.

`psdTolerance` (optional, type = "Sigma") A float. Tolerance (on the smallest eigenvalue) for the PSD check on `M`. Default 1e-8.

## Value

An object of class `"mspca"` (a list) with fields: `x_best` (p x r matrix of sparse PC loadings), `objective_value`, `feasibility_violation`, `runtime`, `variance_explained` (per-PC explained variance), and `total_variance` (trace of the covariance matrix). With `type = "X"` it additionally records `inputType`, `center`, `scale`, `divisor`, `nObs`, and `p`. Use `print()` to display the sparse loadings and `summary()` for a full per-PC breakdown.

## Examples

```
# From a covariance/correlation matrix (the default type):
TestMat <- cor(mtcars)
```

```
res <- mspca(TestMat, r = 2, ks = c(4, 4), verbose = FALSE)
print(res, TestMat)
# Equivalent call from the raw data matrix (C need not be passed to print):
res_X <- mspca(as.matrix(mtcars), r = 2, ks = c(4, 4), type = "X", verbose = FALSE)
print(res_X)
```

---

`print.mspca` *Print an mspca Object*

---

## Description

S3 print method for objects of class `"mspca"` returned by `mspca()`. Displays the sparse loading matrix (restricted to the union of non-zero rows) together with the percentage of variance explained and the number of non-zero loadings per component.

## Usage

```
## S3 method for class 'mspca'
print(x, C = NULL, digits = NULL, ...)
```

## Arguments

| | |
|---|---|
| `x` | An object of class `"mspca"`, as returned by `mspca()`. |
| `C` | (optional) A numeric matrix (p x p). The covariance or correlation matrix used when fitting. May be omitted for `type = "X"` results. |
| `digits` | An integer or `NULL`. Number of significant digits for display. When `NULL` (the default), `getOption("digits")` is used, so the output respects `options(digits = ...)`. |
| `...` | Further arguments required by the `print()` generic; not used by this method. |

## Details

When the model was fit from a covariance/correlation matrix (`type = "Sigma"`), pass that matrix as `C` so that per-PC variance figures can be computed; when it was fit from a raw data matrix (`type = "X"`), `C` may be omitted because the figures are stored inside the object.

## Value

Invisibly returns `x`.

## Examples

```
TestMat <- cor(mtcars)
res <- mspca(TestMat, r = 2, ks = c(4, 4), verbose = FALSE)
print(res, TestMat)
```

summary.mspca *Summarize an mspca Object*

## Description

S3 summary method for objects of class "mspca" returned by mspca(). Returns (and prints) a per-PC summary table (number of non-zero loadings, variance explained, FVE, and cumulative FVE) together with the pairwise feasibility violation matrix and the total solver runtime.

## Usage

```
## S3 method for class 'mspca'
summary(object, C = NULL, feasibilityConstraintType = 0L, digits = NULL, ...)
```

## Arguments

| | |
|---|---|
| object | An object of class "mspca", as returned by mspca(). |
| C | (optional) A numeric matrix (p x p). The covariance or correlation matrix used when fitting. May be omitted for type = "X" results, where the figures are stored inside the object. |
| feasibilityConstraintType | An integer. Type of constraint used to compute the feasibility violation reported in the summary. 0 (default) for orthogonality; 1 for zero pairwise correlation. |
| digits | An integer or NULL. Number of significant digits for display. When NULL (the default), getOption("digits") is used. |
| ... | Further arguments required by the summary() generic; not used by this method. |

## Value

Invisibly returns a list of class "summary.mspca" with fields:

table Data frame with columns PC, nonzero, variance, fve, and cumulative_fve.

feasibility_mat r x r matrix of pairwise feasibility violations ($|u_i^\top u_j|$ or $|u_i^\top C u_j|$). Diagonal and lower triangle are NA.

feasibility Scalar total feasibility violation (sum of the upper triangle of feasibility_mat).

runtime Solver runtime in seconds (if stored in the object).

r Number of sparse PCs.

inputType "Sigma" or "X".

## Examples

```
TestMat <- cor(mtcars)
res <- mspca(TestMat, r = 2, ks = c(4, 4), verbose = FALSE)
summary(res, TestMat)
```

tpm *Truncated Power Method*

## Description

Returns the leading sparse principal component of a dataset using the truncated power method. As in `mspca()`, the data is passed as a single argument `M` whose interpretation is set by `type`: `"Sigma"` (default) for a covariance/correlation matrix (p x p) or `"X"` for a raw data matrix (n x p). See `mspca()` for the raw-data preprocessing controls.

## Usage

```
tpm(
  M,
  k,
  type = c("Sigma", "X"),
  maxIter = 200,
  verbose = TRUE,
  timeLimit = 10,
  center = TRUE,
  scale = FALSE,
  divisor = c("n-1", "n"),
  checkPSD = TRUE,
  symTolerance = 1e-08,
  psdTolerance = 1e-08
)
```

## Arguments

| | |
|---|---|
| `M` | A matrix. The data, interpreted according to `type`: a covariance/ correlation matrix (p x p) when `type = "Sigma"`, or a raw data matrix (n x p) when `type = "X"`. |
| `k` | An integer. Target sparsity of the PC. |
| `type` | (optional) Either "Sigma" (default; `M` is a covariance/correlation matrix) or "X" (`M` is a raw data matrix). |
| `maxIter` | (optional) An integer. Maximum number of iterations of the algorithm. Default 200. |
| `verbose` | (optional) A Boolean. Controls console output. Default TRUE. |
| `timeLimit` | (optional) An integer. Maximum time in seconds. Default 10. |
| `center` | (optional, type = "X") A Boolean. Center the columns of `M`. Default TRUE. |
| `scale` | (optional, type = "X") A Boolean. Scale the columns of `M` to unit variance. Default FALSE. |
| `divisor` | (optional, type = "X") Either "n-1" (default) or "n". |
| `checkPSD` | (optional, type = "Sigma") A Boolean. Verify `M` is PSD. Default TRUE. |
| `symTolerance` | (optional, type = "Sigma") A float. Symmetry-check tolerance. Default 1e-8. |
| `psdTolerance` | (optional, type = "Sigma") A float. PSD-check tolerance. Default 1e-8. |

## Value

An object of class `"tpm"` (a list) with fields: `x_best` (p x 1 matrix containing the sparse PC loading), `objective_value`, and `runtime`. With `type = "X"` it additionally records `inputType`, `center`, `scale`, `divisor`, `nObs`, and `p`.

## References

Yuan, X. T., & Zhang, T. (2013). Truncated power method for sparse eigenvalue problems. *The Journal of Machine Learning Research*, **14**(1), 899–925.

## Examples

```
TestMat <- cor(mtcars)
tpm(TestMat, 4)
```

---

# variance_explained_perPC

*Variance Explained Per PC*

---

## Description

Computes the variance explained by each PC.

## Usage

```
variance_explained_perPC(C, U)
```

## Arguments

| | |
|---|---|
| `C` | A matrix. The correlation or covariance matrix (p x p). |
| `U` | A matrix. The matrix containing the r PCs (p x r). |

## Value

A numeric vector of length `ncol(U)`.

# Index