Package: scPCA (via r-universe)

October 23, 2026

Title Sparse Contrastive Principal Component Analysis

Version 1.27.0

Description A toolbox for sparse contrastive principal component analysis (scPCA) of high-dimensional biological data. scPCA combines the stability and interpretability of sparse PCA with contrastive PCA's ability to disentangle biological signal from unwanted variation through the use of control data. Also implements and extends cPCA.

Depends R (>= 4.0.0)

Imports stats, methods, assertthat, tibble, dplyr, purrr, stringr, Rdpack, matrixStats, BiocParallel, elasticnet, sparsepca, cluster, kernlab, origami, RSpectra, coop, Matrix, DelayedArray, ScaledMatrix, MatrixGenerics

Suggests DelayedMatrixStats, sparseMatrixStats, testthat (>= 2.1.0), covr, knitr, rmarkdown, BiocStyle, ggplot2, ggpubr, splatter, SingleCellExperiment, microbenchmark

License MIT + file LICENSE

URL https://github.com/PhilBoileau/scPCA

BugReports https://github.com/PhilBoileau/scPCA/issues

Encoding UTF-8

LazyData true

VignetteBuilder knitr

RoxygenNote 7.3.1

RdMacros Rdpack

biocViews PrincipalComponent, GeneExpression, DifferentialExpression, Sequencing, Microarray, RNASeq

Config/pak/sysreqs libicu-dev zlib1g-dev

Repository https://bioc.r-universe.dev

Date/Publication 2026-04-28 12:51:20 UTC

RemoteUrl https://github.com/bioc/scPCA

RemoteRef HEAD

RemoteSha 137ceaa60750e1666212fcaf481ad786f2c00632

Contents

background_df 2
scPCA 3
toy_df 6

Index **8**

`background_df` *Simulated Background Data for cPCA and scPCA*

Description

The background data consisting of 400 observations and 30 variables was simulated as follows:

- Each of the first 10 variables was drawn from $N(0, 10)$
- Variables 11 through 20 were drawn from $N(0, 3)$
- Variables 21 through 30 were drawn from $N(0, 1)$

Usage

```
data(background_df)
```

Format

A simple `data.frame`.

Examples

```
data(background_df)
```

scPCA *Sparse Contrastive Principal Component Analysis*

Description

Given target and background data frames or matrices, scPCA will perform the sparse contrastive principal component analysis (scPCA) of the target data for a given number of eigenvectors, a vector of real-valued contrast parameters, and a vector of sparsity inducing penalty terms.

If instead you wish to perform contrastive principal component analysis (cPCA), set the penalties argument to 0. So long as the n_centers parameter is larger than one, the automated hyperparameter tuning heuristic described in Boileau et al. (2020) is used. Otherwise, the semi-automated approach of Abid et al. (2018) is used to select the appropriate hyperparameter.

Usage

```
scPCA(
  target,
  background,
  center = TRUE,
  scale = FALSE,
  n_eigen = 2,
  cv = NULL,
  alg = c("iterative", "var_proj", "rand_var_proj"),
  contrasts = exp(seq(log(0.1), log(1000), length.out = 40)),
  penalties = seq(0.05, 1, length.out = 20),
  clust_method = c("kmeans", "pam", "hclust"),
  n_centers = NULL,
  max_iter = 10,
  linkage_method = "complete",
  n_medoids = 8,
  parallel = FALSE,
  clusters = NULL,
  eigdecomp_tol = 1e-10,
  eigdecomp_iter = 1000,
  scaled_matrix = FALSE
)
```

Arguments

target	The target (experimental) data set, in a standard format such as a data.frame or matrix. dgCMatrix and DelayedMatrix objects are also supported.
background	The background data set, in a standard format such as a data.frame or matrix. The features must match the features of the target data set. dgCMatrix and DelayedMatrix objects are also supported.
center	A logical indicating whether the target and background data sets' features should be centered to mean zero.

scale — A logical indicating whether the target and background data sets' features should be scaled to unit variance.

n_eigen — A numeric indicating the number of eigenvectors (or (sparse) contrastive components) to be computed. Two eigenvectors are computed by default.

cv — A numeric indicating the number of cross-validation folds to use in choosing the optimal contrastive and penalization parameters from over the grids of contrasts and penalties. Cross-validation is expected to improve the robustness and generalization of the choice of these parameters. However, it increases the time the procedure costs. The default is therefore NULL, corresponding to no cross-validation.

alg — A character indicating the sparse PCA algorithm used to sparsify the contrastive loadings. Currently supports iterative for the Zou et al. (2006) implementation, var_proj for the non-randomized Erichson et al. (2018) solution, and rand_var_proj for the randomized Erichson et al. (2018) implementation. Defaults to iterative.

contrasts — A numeric vector of the contrastive parameters. Each element must be a unique, non-negative real number. By default, 40 logarithmically spaced values between 0.1 and 1000 are used. If a single value is provided and penalties is set to 0, then n_centers, clust_method, max_iter, linkage_method, n_medoids, and parallel can be safely ignored.

penalties — A numeric vector of the L1 penalty terms on the loadings. The default is to use 20 equidistant values between 0.05 and 1. If penalties is set to 0, then cPCA is performed in place of scPCA. See contrasts and n_centers arguments for more infotmation.

clust_method — A character specifying the clustering method to use for choosing the optimal contrastive parameter. Currently, this is limited to either k-means, partitioning around medoids (PAM), and hierarchical clustering. The default is k-means clustering.

n_centers — A numeric giving the number of centers to use in the clustering algorithm. If set to 1, cPCA, as first proposed by Abid et al. (2018), is performed, regardless of what the penalties argument is set to.

max_iter — A numeric giving the maximum number of iterations to be used in k-means clustering. Defaults to 10.

linkage_method — A character specifying the agglomerative linkage method to be used if clust_method = "hclust". The options are ward.D2, single, complete, average, mcquitty, median, and centroid. The default is complete.

n_medoids — A numeric indicating the number of medoids to consider if n_centers is set to 1 and contrasts is a vector of length 2 or more. The default is 8 medoids.

parallel — A logical indicating whether to invoke parallel processing via the **BiocParallel** infrastructure. The default is FALSE for sequential evaluation.

clusters — A numeric vector of cluster labels for observations in the target data. Defaults to NULL, but is otherwise used to identify the optimal set of hyperparameters when fitting the scPCA and the automated version of cPCA. If a vector is provided, the n_centers, clust_method, max_iter, linkage_method, and n_medoids arguments can be safely ignored.

`eigdecomp_tol` A `numeric` providing the level of precision used by eigendecompositon calculations. Defaults to `1e-10`.

`eigdecomp_iter` A `numeric` indicating the maximum number of interations performed by eigendecompositon calculations. Defaults to `1000`.

`scaled_matrix` A `logical` indicating whether to output a `ScaledMatrix` object. The centering and scaling procedure is delayed until later, permitting more efficient matrix multiplication and row or column sums downstream. However, this comes at the at the cost of numerical precision. Defaults to `FALSE`.

Value

A list containing the following components:

- `rotation`: The matrix of variable loadings if `n_centers` is larger than one. Otherwise, a list of rotation matrices is returned, one for each medoid. The number of medoids is specified by `n_medoids`.
- `x`: The rotated data, centred and scaled if requested, multiplied by the rotation matrix if `n_centers` is larger than one. Otherwise, a list of rotated data matrices is returned, one for each medoid. The number of medoids is specified by `n_medoids`.
- contrast: The optimal contrastive parameter.
- penalty: The optimal L1 penalty term.
- center: A logical indicating whether the target dataset was centered.
- scale: A logical indicating whether the target dataset was scaled.

References

Abid A, Zhang MJ, Bagaria VK, Zou J (2018). "Exploring patterns enriched in a dataset with contrastive principal component analysis." *Nature communications*, **9**(1), 2134.

Boileau P, Hejazi NS, Dudoit S (2020). "Exploring High-Dimensional Biological Data with Sparse Contrastive Principal Component Analysis." *Bioinformatics*. ISSN 1367-4803. doi:10.1093/bioinformatics/btaa176. btaa176, https://academic.oup.com/bioinformatics/article-pdf/doi/10.1093/bioinformatics/btaa176/32914142/btaa1

Erichson NB, Zeng P, Manohar K, Brunton SL, Kutz JN, Aravkin AY (2018). "Sparse Principal Component Analysis via Variable Projection." *ArXiv*, **abs/1804.00341**.

Zou H, Hastie T, Tibshirani R (2006). "Sparse principal component analysis." *Journal of computational and graphical statistics*, **15**(2), 265–286.

Examples

```
# perform cPCA on the simulated data set
scPCA(
  target = toy_df[, 1:30],
  background = background_df,
  contrasts = exp(seq(log(0.1), log(100), length.out = 5)),
  penalties = 0,
  n_centers = 4
```

```
)

# perform scPCA on the simulated data set
scPCA(
  target = toy_df[, 1:30],
  background = background_df,
  contrasts = exp(seq(log(0.1), log(100), length.out = 5)),
  penalties = seq(0.1, 1, length.out = 3),
  n_centers = 4
)

# perform cPCA on the simulated data set with known clusters
scPCA(
  target = toy_df[, 1:30],
  background = background_df,
  contrasts = exp(seq(log(0.1), log(100), length.out = 5)),
  penalties = 0,
  clusters = toy_df[, 31]
)

# cPCA as implemented in Abid et al.
scPCA(
  target = toy_df[, 1:30],
  background = background_df,
  contrasts = exp(seq(log(0.1), log(100), length.out = 10)),
  penalties = 0,
  n_centers = 1
)
```

---

toy_df *Simulated Target Data for cPCA and scPCA*

Description

The toy data consisting of 400 observations and 31 variables was simulated as follows:

- Each of the first 10 variables was drawn from $N(0, 10)$
- For group 1 and 2, variables 11 through 20 were drawn from $N(0, 1)$
- For group 3 and 4, variables 11 through 20 were drawn from $N(3, 1)$
- For group 1 and 3, variables 21 though 30 were drawn from $N(-3, 1)$
- For group 2 and 4, variables 21 though 30 were drawn from $N(0, 1)$
- The last column provides each observations group number

Usage

```
data(toy_df)
```

Format

A simple `data.frame`.

Examples

```
data(toy_df)
```

Index

∗ **datasets**
background_df, 2
toy_df, 6

background_df, 2

ScaledMatrix, *5*
scPCA, 3

toy_df, 6