Package ‘pcaL1’

May 3, 2025

Version 1.5.9

Date 2025-04-25

Title L1-Norm PCA Methods

License GPL (>= 3)

Description Implementations of several methods for principal component analysis using the L1 norm. The package depends on COIN-OR Clp version >= 1.17.4. The methods implemented are
PCA-L1 (Kwak 2008) <DOI:10.1109/TPAMI.2008.114>,
L1-PCA (Ke and Kanade 2003, 2005) <DOI:10.1109/CVPR.2005.309>,
L1-PCA* (Brooks, Dula, and Boone 2013) <DOI:10.1016/j.csda.2012.11.007>,
L1-PCAhp (Visentin, Prestwich and Armagan 2016) <DOI:10.1007/978-3-319-46227-1_37>,
wPCA (Park and Klabjan 2016) <DOI:10.1109/ICDM.2016.0054>,
awPCA (Park and Klabjan 2016) <DOI:10.1109/ICDM.2016.0054>,
PCA-Lp (Kwak 2014) <DOI:10.1109/TCYB.2013.2262936>, and
SharpEl1-PCA (Brooks and Dula, submitted).

SystemRequirements COIN-OR Clp (>= 1.17.4)

NeedsCompilation yes

Author Sapan Jot [aut],
Paul Brooks [aut, cre],
Andrea Visentin [aut],
Young Woong Park [aut],
and Yi-Hui Zhou [aut]

Maintainer Paul Brooks <jpbrooks@vcu.edu>

Repository CRAN

Date/Publication 2025-05-03 19:30:02 UTC

Contents

`pcaL1-package` *pcaL1: L1-Norm PCA Methods*

Description

This package contains implementations of six principal component analysis methods using the L1 norm. The package depends on COIN-OR Clp version >= 1.17.4. The methods implemented are PCA-L1 (Kwak 2008), L1-PCA (Ke and Kanade 2003, 2005), L1-PCA* (Brooks, Dula, and Boone 2013), L1-PCAhp (Visentin, Prestwich and Armagan 2016), wPCA (Park and Klabjan 2016), and awPCA (Park and Klabjan 2016).

Details

Package:	pcaL1
Version:	1.5.9
Date:	2025-04-25
License:	GPL (>=3)
URL:	http://www.optimization-online.org/DB_HTML/2012/04/3436.html, http://www.coin-or.org
SystemRequirements:	COIN-OR Clp (>= 1.17.4)

Index:

awl1pca	awPCA
l1pca	L1-PCA
l1pcahp	L1-PCAhp
l1pcastar	L1-PCA*
l1projection	L1-Norm Projection on a Subspace
L2PCA_approx	Subroutine for awl1pca
l2projection	L2-Norm Projection on a Subspace
pcal1	PCA-L1
pcalp	PCA-Lp
pcaL1-package	pcaL1: L1-Norm PCA Methods
plot.awl1pca	Plot an awl1pca Object
plot.l1pca	Plot an l1pca Object
plot.l1pcahp	Plot an l1pcahp Object
plot.l1pcastar	Plot an l1pcastar Object
plot.pcal1	Plot a pcal1 Object
plot.pcalp	Plot a pcalp Object
plot.wl1pca	Plot an wl1pca Object
plot.sharpel1pca	Plot a sharpel1pca Object
sharpel1pca	SharpeEL1-PCA
sharpel1rs	SharpEl1-RS
sparsel1pca	SparseEl1-PCA
wl1pca	wPCA

Author(s)

Sapan Jot <sapan.madaan@gmail.com>, Paul Brooks <jpbrooks@vcu.edu>, Andrea Visentin <andrea.visentin@insight-centre.org>,Young Woong Park <ywpark@iastate.edu>, and Yi-Hui Zhou <yihui_zhou@ncsu.edu>

Maintainer: Paul Brooks <jpbrooks@vcu.edu>

References

1. Brooks and Dula (2017) Estimating L1-Norm Best-Fit Lines, submitted
2. Brooks J.P., Dula J.H., and Boone E.L. (2013) A Pure L1-Norm Princpal Component Analysis, *Computational Statistics & Data Analysis*, 61:83-98. DOI:10.1016/j.csda.2012.11.007
3. Ke Q. and Kanade T. (2005) Robust L1 Norm Factorization in the Presence of Outliers and Missing Data by Alternative Convex Programming, *IEEE Conference on Computer Vision and Pattern Recognition*. DOI:10.1109/CVPR.2005.309
4. Kwak N. (2008) Principal Component Analysis Based on L1-Norm Maximization, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30: 1672-1680. DOI:10.1109/TPAMI.2008.114
5. Kwak N. (2014) Principal Component Analysis by Lp-Norm Maximization, *IEEE Transactions on Cybernetics*, 44:594-609. DOI:10.1109/TCYB.2013.2262936
6. Park, Y.W. and Klabjan, D. (2016) Iteratively Reweighted Least Squares Algorithms for L1-Norm Principal Component Analysis, *IEEE International Conference on Data Mining (ICDM)*. DOI: 10.1109/ICDM.2016.0054
7. Visentin A., Prestwich S., and Armagan S. T. (2016) Robust Principal Component Analysis by Reverse Iterative Linear Programming, *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 593-605. DOI:10.1007/978-3-319-46227-1_37

8. Zhou, Y.-H. and Marron, J.S. (2016) Visualization of Robust L1PCA, *Stat*, 5:173-184. DOI:10.1002/sta4.113

`awl1pca` *awPCA*

Description

Performs a principal component analysis using the algorithm awPCA described by Park and Klabjan (2016).

Usage

```
awl1pca(X, projDim=1, center=TRUE, projections="l2",
        tolerance=0.001, iterations=200, beta=0.99, gamma=0.1)
```

Arguments

`X`	data, must be in `matrix` or table form.
`projDim`	number of dimensions to project data into, must be an integer, default is 1.
`center`	whether to center the data using the mean, default is TRUE.
`projections`	whether to calculate projections (reconstructions and scores) using the L2 norm ("l2", default) or the L1 norm ("l1").
`tolerance`	for testing convergence; if the sum of absolute values of loadings vectors is smaller, then the algorithm terminates.
`iterations`	maximum number of iterations in optimization routine.
`beta`	algorithm parameter to set up bound for weights.
`gamma`	algorithm parameter to determine whether to use approximation formula or prcomp function.

Details

The calculation is performed according to the algorithm described by Park and Klabjan (2016). The method is an iteratively reweighted least squares algorithm for L1-norm principal component analysis.

Value

'awl1pca' returns a list with class "awl1pca" containing the following components:

`loadings`	the matrix of variable loadings. The matrix has dimension ncol(X) x projDim. The columns define the projected subspace.
`scores`	the matrix of projected points. The matrix has dimension nrow(X) x projDim.
`projPoints`	the matrix of L2-norm projections of points on the fitted subspace in terms of the original coordinates. The matrix has dimension nrow(X) x ncol(X).
`L1error`	sum of the L1 norm of reconstruction errors.
`nIter`	number of iterations.
`ElapsedTime`	elapsed time.

References

Park, Y.W. and Klabjan, D. (2016) Iteratively Reweighted Least Squares Algorithms for L1-Norm Principal Component Analysis, *IEEE International Conference on Data Mining (ICDM)*, 2016. DOI: 10.1109/ICDM.2016.0054

Examples

```
##for 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100) +
              matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
myawl1pca <- awl1pca(X)

##projects data into 2 dimensions.
myawl1pca <- awl1pca(X, projDim=2, center=FALSE)

## plot first two scores
plot(myawl1pca$scores)
```

`l1pca` *L1-PCA*

Description

Performs a principal component analysis using the algorithm L1-PCA given by Ke and Kanade (2005).

Usage

```
   l1pca(X, projDim=1, center=TRUE, projections="l1",
 initialize="l2pca", tolerance=0.0001, iterations=10)
```

Arguments

`X`	data, must be in `matrix` or table form.
`projDim`	number of dimensions to project data into, must be an integer, default is 1.
`center`	whether to center the data using the median, default is TRUE.
`projections`	Whether to calculate reconstructions and scores using the L1 ("l1", default) or L2 ("l2") norm.
`initialize`	initial guess for loadings matrix. Options are: "l2pca" - use traditional PCA/SVD, "random" - use a randomly-generated matrix. The user can also provide a matrix as an initial guess.
`tolerance`	sets the convergence tolerance for the algorithm, default is 0.0001.
`iterations`	sets the number of iterations to run before returning the result, default is 10.

Details

The calculation is performed according to the linear programming-based algorithm described by Ke and Kanade (2005). The method is a locally-convergent algorithm for finding the L1-norm best-fit subspace by alternatively optimizing the scores and the loadings matrix at each iteration. Linear programming instances are solved using Clp (http://www.coin-or.org)

Value

'l1pca' returns a list with class "l1pca" containing the following components:

`loadings` the matrix of variable loadings. The matrix has dimension ncol(X) x projDim. The columns defined the projected subspace.

`scores` the matrix of projected points. The matrix has dimension nrow(X) x projDim.

`dispExp` the proportion of L1 dispersion explained by the loadings vectors. Calculated as the L1 dispersion of the score on each component divided by the L1 dispersion in the original data.

`projPoints` the matrix of projected points in terms of the original coordinates (reconstructions). The matrix has dimension nrow(X) x ncol(X).

References

Ke Q. and Kanade T. (2005) Robust L1 norm factorization in the presence of outliers and missing data by alternative convex programming, *IEEE Conference on Computer Vision and Pattern Recognition*. DOI:10.1109/CVPR.2005.309

Examples

```
##for 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100) +
                matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
myl1pca <- l1pca(X)

##projects data into 2 dimensions.
myl1pca <- l1pca(X, projDim=2, center=FALSE,
                tolerance=0.00001, iterations=20)

## plot first two scores
plot(myl1pca$scores)
```

`l1pcahp` *L1-PCAhp*

Description

Performs a principal component analysis using the algorithm L1-PCAhp described by Visentin, Prestwich and Armagan (2016)

Usage

```
l1pcahp(X, projDim=1, center=TRUE, projections="none",
        initialize="l2pca", threshold=0.0001)
```

Arguments

`X`	data, must be in `matrix` or table form.
`projDim`	number of dimensions to project data into, must be an integer, default is 1.
`center`	whether to center the data using the median, default is TRUE.
`projections`	whether to calculate reconstructions and scores using the L1 norm ("l1") the L2 norm ("l2") or not at all ("none", default).
`initialize`	method for initial guess for loadings matrix. Options are: "l2pca" - use traditional PCA/SVD, "random" - use a randomly-generated matrix.
`threshold`	sets the convergence threshold for the algorithm, default is 0.001.

Details

The calculation is performed according to the algorithm described by Visentin, Prestwich and Armagan (2016). The algorithm computes components iteratively in reverse, using a new heuristic based on Linear Programming. Linear programming instances are solved using Clp (http://www.coin-or.org).

Value

'l1pcahp' returns a list with class "l1pcahp" containing the following components:

`loadings`	the matrix of variable loadings. The matrix has dimension ncol(X) x ncol(X). The columns define the projected subspace.
`scores`	the matrix of projected points. The matrix has dimension nrow(X) x projDim.
`dispExp`	the proportion of L1 dispersion explained by the loadings vectors. Calculated as the L1 dispersion of the score on each component divided by the L1 dispersion in the original data.
`projPoints`	the matrix of projected points in terms of the original coordinates. The matrix has dimension nrow(X) x ncol(X).

References

Visentin A., Prestwich S., and Armagan S. T. (2016) Robust Principal Component Analysis by Reverse Iterative Linear Programming, *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 593-605. DOI:10.1007/978-3-319-46227-1_37

Examples

```
##for a 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100) +
```

```
                matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
myl1pcahp <- l1pcahp(X)

##projects data into 2 dimensions.
myl1pcahp <- l1pcahp(X, projDim=2, center=FALSE, projections="l1")

## plot first two scores
plot(myl1pcahp$scores)
```

`l1pcastar` *L1-PCA**

Description

Performs a principal component analysis using the algorithm L1-PCA* described by Brooks, Dula, and Boone (2013)

Usage

```
l1pcastar(X, projDim=1, center=TRUE, projections="none")
```

Arguments

`X`	data, must be in `matrix` or table form
`projDim`	number of dimensions to project data into, must be an integer, default is 1
`center`	whether to center the data using the median, default is TRUE
`projections`	whether to calculate reconstructions and scores using the L1 norm ("l1") the L2 norm ("l2") or not at all ("none", default)

Details

The calculation is performed according to the algorithm described by Brooks, Dula, and Boone (2013). The algorithm finds successive directions of minimum dispersion in the data by finding the L1-norm best-fit hyperplane at each iteration. Linear programming instances are solved using Clp (http://www.coin-or.org)

Value

'l1pcastar' returns a list with class "l1pcastar" containing the following components:

`loadings`	the matrix of variable loadings. The matrix has dimension ncol(X) x ncol(X). The columns define the projected subspace.
`scores`	the matrix of projected points. The matrix has dimension nrow(X) x projDim.
`dispExp`	the proportion of L1 dispersion explained by the loadings vectors. Calculated as the L1 dispersion of the score on each component divided by the L1 dispersion in the original data.
`projPoints`	the matrix of projected points in terms of the original coordinates. The matrix has dimension nrow(X) x ncol(X).

References

1. Brooks J.P., Dula J.H., and Boone E.L. (2013) A Pure L1-Norm Princpal Component Analysis, *Computational Statistics & Data Analysis*, 61:83-98. DOI:10.1016/j.csda.2012.11.007
2. Zhou, Y.-H. and Marron, J.S. (2016) Visualization of Robust L1PCA, *Stat*, 5:173-184. DOI:10.1002/sta4.113

Examples

```
##for a 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100) +
                 matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
myl1pcastar <- l1pcastar(X)

##projects data into 2 dimensions.
myl1pcastar <- l1pcastar(X, projDim=2, center=FALSE, projections="l1")

## plot first two scores
plot(myl1pcastar$scores)
```

`l1projection` *L1 Projection*

Description

Provides the L1-norm projection of points on a subspace, including both scores and reconstructions.

Usage

```
l1projection(X, loadings)
```

Arguments

`X`	data, in `matrix` or table form
`loadings`	an orthonormal matrix of loadings vectors

Details

The scores and reconstructions are calculated by solving a linear program.

Value

'l1projection' returns a list containing the following components:

`scores`	the matrix of projected points
`projPoints`	the matrix of projected points in terms of the original coordinates (reconstructions)

L2PCA_approx *L2PCA_approx*

Description

Provides an approximation of traditional PCA described by Park and Klabjan (2016) as a subroutine for awl1pca.

Usage

```
L2PCA_approx(ev.prev, pc.prev, projDim, X.diff)
```

Arguments

ev.prev	matrix of principal component loadings from a previous iteration of awl1pca
pc.prev	vector of eigenvalues from previous iteration of awl1pca
projDim	number of dimensions to project data into, must be an integer
X.diff	The difference between the current weighted matrix estimate and the estimate from the previous iteration

Details

The calculation is performed according to equations (11) and (12) in Park and Klabjan (2016). The method is an approximation for traditional principal component analysis.

Value

'L2PCA_approx' returns a list containing the following components:

eigenvalues	Estimate of eigenvalues of the covariance matrix.
eigenvectors	Estimate of eigenvectors of the covariance matrix.

References

Park, Y.W. and Klabjan, D. (2016) Iteratively Reweighted Least Squares Algorithms for L1-Norm Principal Component Analysis, *IEEE International Conference on Data Mining (ICDM)*, 2016.

See Also

awl1pca

l2projection — *L2 Projection*

Description

Provides the L2-norm projection of points on a subspace, including both scores and reconstructions.

Usage

```
l2projection(X, loadings)
```

Arguments

`X`	data, in `matrix` or table form
`loadings`	an orthonormal matrix of loadings vectors

Details

The scores and reconstructions are calculated by solving a linear program.

Value

'l2projection' returns a list containing the following components:

`scores`	the matrix of projected points
`projPoints`	the matrix of projected points in terms of the original coordinates (reconstructions)

pcal1 — *PCA-L1*

Description

Performs a principal component analysis using the algorithm PCA-L1 given by Kwak (2008).

Usage

```
pcal1(X, projDim=1, center=TRUE, projections="none", initialize="l2pca")
```

Arguments

`X`	data, must be in `matrix` or table form.
`projDim`	number of dimensions to project data into, must be an integer, default is 1.
`center`	whether to center the data using the median, default is TRUE.
`projections`	whether to calculate reconstructions and scores using the L1 norm ("l1") the L2 norm ("l2") or not at all ("none", default).
`initialize`	initial guess for first component. Options are: "l2pca" - use traditional PCA/SVD, "maxx" - use the point with the largest norm, "random" - use a random vector. The user can also provide a vector as the initial guess.

Details

The calculation is performed according to the algorithm described by Kwak (2008). The method is a locally-convergent algorithm for finding successive directions of maximum L1 dispersion.

Value

'pcal1' returns a list with class "pcal1" containing the following components:

`loadings`	the matrix of variable loadings. The matrix has dimension ncol(X) x projDim. The columns define the projected subspace.
`scores`	the matrix of projected points. The matrix has dimension nrow(X) x projDim.
`dispExp`	the proportion of L1 dispersion explained by the loadings vectors. Calculated as the L1 dispersion of the score on each component divided by the L1 dispersion in the original data.
`projPoints`	the matrix of projected points in terms of the original coordinates (reconstructions). The matrix has dimension nrow(X) x ncol(X).

References

Kwak N. (2008) Principal component analysis based on L1-norm maximization, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30: 1672-1680. DOI:10.1109/TPAMI.2008.114

Examples

```
##for 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100) +
                matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
mypcal1 <- pcal1(X)

##projects data into 2 dimensions.
mypcal1 <- pcal1(X, projDim=2, center=FALSE, projections="l1")

## plot first two scores
plot(mypcal1$scores)
```

pcalp *PCA-Lp*

Description

Performs a principal component analysis using the greedy algorithms PCA-Lp(G) and PCA-Lp(L) given by Kwak (2014).

Usage

```
pcalp(X, projDim=1, p = 1.0, center=TRUE, projections="none",
        initialize="l2pca",solution = "L",
epsilon = 0.0000000001, lratio = 0.02)
```

Arguments

`X`	data, must be in `matrix` or table form.
`projDim`	number of dimensions to project data into, must be an integer, default is 1.
`p`	p-norm use to measure the distance between points.
`center`	whether to center the data using the median, default is TRUE.
`projections`	whether to calculate reconstructions and scores using the L1 norm ("l1") the L2 norm ("l2") or not at all ("none", default).
`initialize`	method for initial guess for component. Options are: "l2pca" - use traditional PCA/SVD, "maxx" - use the point with the largest norm, "random" - use a random vector.
`solution`	method projection vector update. Options are: "G" - PCA-Lp(G) implementation: Gradient search, "L" - PCA-Lp(L) implementation: Lagrangian (default).
`epsilon`	for checking convergence.
`lratio`	learning ratio, default is 0.02. Suggested value 1/(nr. instances).

Details

The calculation is performed according to the algorithm described by Kwak (2014), an extension of the original Kwak(2008). The method is a greedy locally-convergent algorithm for finding successive directions of maximum Lp dispersion.

Value

'pcalp' returns a list with class "pcalp" containing the following components:

`loadings`	the matrix of variable loadings. The matrix has dimension ncol(X) x projDim. The columns define the projected subspace.
`scores`	the matrix of projected points. The matrix has dimension nrow(X) x projDim.

dispExp	the proportion of L1 dispersion explained by the loadings vectors. Calculated as the L1 dispersion of the score on each component divided by the L1 dispersion in the original data.
projPoints	the matrix of projected points in terms of the original coordinates. The matrix has dimension nrow(X) x ncol(X).

References

Kwak N. (2008) Principal component analysis based on L1-norm maximization, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30: 1672-1680. DOI:10.1109/TPAMI.2008.114

Kwak N. (2014). Principal component analysis by Lp-norm maximization. *IEEE transactions on cybernetics, 44(5)*, 594-609. DOI: 10.1109/TCYB.2013.2262936

Examples

```
##for 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100)
               + matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
mypcalp <- pcalp(X, p = 1.5)

##projects data into 2 dimensions.
mypcalp <- pcalp(X, projDim=2, p = 1.5, center=FALSE, projections="l1")

## plot first two scores
plot(mypcalp$scores)
```

`plot.awl1pca` *Plot an awl1pca Object*

Description

Plots the scores on the first two principal components.

Usage

```
## S3 method for class 'awl1pca'
plot(x, ...)
```

Arguments

x	an object of class `awl1pca` with scores for at least the first two dimensions
...	arguments to be passed to or from other methods.

Details

This function is a method for the generic function `plot`, for objects of class `awl1pca`.

See Also

l1pcastar

Examples

```
##for a 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100)
              + matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
myawl1pca <- awl1pca(X)

##projects data into 2 dimensions.
myawl1pca <- awl1pca(X, projDim=2, center=FALSE)

## plot first two scores
plot(myawl1pca$scores)
```

---

plot.l1pca *Plot an L1pca Object*

Description

Plots the scores on the first two principal components.

Usage

```
## S3 method for class 'l1pca'
plot(x, ...)
```

Arguments

x	an object of class l1pca with scores for at least the first two dimensions
...	arguments to be passed to or from other methods.

Details

This function is a method for the generic function `plot`, for objects of class `l1pca`.

See Also

l1pca

Examples

```
##for a 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100)
              + matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
myl1pca <- l1pca(X)

##projects data into 2 dimensions.
myl1pca <- l1pca(X, projDim=2, center=FALSE)

## plot first two scores
plot(myl1pca$scores)
```

---

plot.l1pcahp *Plot an L1PCAhp Object*

Description

Plots the scores on the first two principal components.

Usage

```
## S3 method for class 'l1pcahp'
plot(x, ...)
```

Arguments

x	an object of class l1pcahp with scores for at least the first two dimensions
...	arguments to be passed to or from other methods.

Details

This function is a method for the generic function `plot`, for objects of class `l1pcahp`.

See Also

`l1pcastar`

Examples

```
##for a 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100)
               + matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
myl1pcahp <- l1pcahp(X)

##projects data into 2 dimensions.
```

```
myl1pcahp <- l1pcahp(X, projDim=2, center=FALSE, projections="l1")

## plot first two scores
plot(myl1pcahp$scores)
```

---

plot.l1pcastar *Plot an L1pcastar Object*

Description

Plots the scores on the first two principal components.

Usage

```
## S3 method for class 'l1pcastar'
plot(x, ...)
```

Arguments

x	an object of class `l1pcastar` with scores for at least the first two dimensions
...	arguments to be passed to or from other methods.

Details

This function is a method for the generic function `plot`, for objects of class `l1pcastar`.

See Also

`l1pcastar`

Examples

```
##for a 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100)
               + matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
myl1pcastar <- l1pcastar(X)

##projects data into 2 dimensions.
myl1pcastar <- l1pcastar(X, projDim=2, center=FALSE, projections="l1")

## plot first two scores
plot(myl1pcastar$scores)
```

plot.pcal1 *Plot a Pcal1 Object*

Description

Plots the scores on the first two principal components.

Usage

```
## S3 method for class 'pcal1'
plot(x, ...)
```

Arguments

x	an object of class pcal1 with scores for at least the first two dimensions
...	arguments to be passed to or from other methods.

Details

This function is a method for the generic function `plot`, for objects of class `pcal1`.

See Also

`pcal1`

Examples

```
##for a 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100)
               + matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
mypcal1 <- pcal1(X)

##projects data into 2 dimensions.
mypcal1 <- pcal1(X, projDim=2, center=FALSE, projections="l1")

## plot first two scores
plot(mypcal1$scores)
```

plot.pcalp *Plot a Pcalp Object*

Description

Plots the scores on the first two principal components.

Usage

```
## S3 method for class 'pcalp'
plot(x, ...)
```

Arguments

x	an object of class pcalp with scores for at least the first two dimensions
...	arguments to be passed to or from other methods.

Details

This function is a method for the generic function `plot`, for objects of class `pcalp`.

See Also

`pcalp`

Examples

```
##for a 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100)
               + matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
mypcalp <- pcalp(X)

##projects data into 2 dimensions.
mypcalp <- pcalp(X, projDim=2, center=FALSE, projections="l1")

## plot first two scores
plot(mypcalp$scores)
```

plot.sharpel1pca *Plot a Sharpel1pca Object*

Description

Plots the scores on the first two principal components.

Usage

```
## S3 method for class 'sharpel1pca'
plot(x, ...)
```

Arguments

x	an object of class sharpel1pca with scores for at least the first two dimensions
...	arguments to be passed to or from other methods.

Details

This function is a method for the generic function plot, for objects of class sharpel1pca.

See Also

sharpel1pca

Examples

```
##for a 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100)
               + matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
mysharpel1pca <- sharpel1pca(X)

##projects data into 2 dimensions.
mysharpel1pca <- sharpel1pca(X, projDim=2, center=FALSE, projections="l1")

## plot first two scores
plot(mysharpel1pca$scores)
```

plot.sharpel1rs — *Plot a Sharpel1rs Object*

Description

Plots the scores on the first two principal components.

Usage

```
## S3 method for class 'sharpel1rs'
plot(x, ...)
```

Arguments

x	an object of class sharpel1rs with scores for at least the first two dimensions
...	arguments to be passed to or from other methods.

Details

This function is a method for the generic function `plot`, for objects of class `sharpel1rs`.

See Also

sharpel1rs

Examples

```
##for a 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100)
               + matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
mysharpel1rs <- sharpel1rs(X)

##projects data into 2 dimensions.
mysharpel1rs <- sharpel1rs(X, projDim=2, center=FALSE, projections="l1")

## plot first two scores
plot(mysharpel1rs$scores)
```

`plot.sparsel1pca` *Plot a Sparsel1pca Object*

Description

Plots the scores on the first two principal components.

Usage

```
## S3 method for class 'sparsel1pca'
plot(x, ...)
```

Arguments

`x`	an object of class `sparsel1pca` with scores for at least the first two dimensions
`...`	arguments to be passed to or from other methods.

Details

This function is a method for the generic function `plot`, for objects of class `sparsel1pca`.

See Also

`sparsel1pca`

Examples

```
##for a 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100)
               + matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
mysparsel1pca <- sparsel1pca(X)

##projects data into 2 dimensions.
mysparsel1pca <- sparsel1pca(X, projDim=2, center=FALSE, projections="l1")

## plot first two scores
plot(mysparsel1pca$scores)
```

`plot.wl1pca` *Plot a Wl1pca Object*

Description

Plots the scores on the first two principal components.

Usage

```
## S3 method for class 'wl1pca'
plot(x, ...)
```

Arguments

`x`	an object of class `wl1pca` with scores for at least the first two dimensions
`...`	arguments to be passed to or from other methods.

Details

This function is a method for the generic function `plot`, for objects of class `wl1pca`.

See Also

`l1pcastar`

Examples

```
##for a 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100)
               + matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
mywl1pca <- wl1pca(X)

##projects data into 2 dimensions.
mywl1pca <- wl1pca(X, projDim=2, center=FALSE)

## plot first two scores
plot(mywl1pca$scores)
```

`sharpel1pca` *SharpEl1-PCA*

Description

Performs a principal component analysis using the algorithm SharpEl1-PCA described by Brooks and Dula (2017, submitted)

Usage

```
sharpel1pca(X, projDim=1, center=TRUE, projections="none")
```

Arguments

`X`	data, must be in `matrix` or table form.
`projDim`	number of dimensions to project data into, must be an integer, default is 1.
`center`	whether to center the data using the median, default is TRUE.
`projections`	whether to calculate reconstructions and scores using the L1 norm ("l1") the L2 norm ("l2") or not at all ("none", default).

Details

The calculation is performed according to the algorithm described by Brooks and Dula (2017, submitted). The algorithm finds successive, orthogonal fitted lines in the data.

Value

'sharpel1pca' returns a list with class "sharpel1pca" containing the following components:

`loadings`	the matrix of variable loadings. The matrix has dimension ncol(X) x projDim. The columns define the projected subspace.
`scores`	the matrix of projected points. The matrix has dimension nrow(X) x projDim.
`dispExp`	the proportion of L1 dispersion explained by the loadings vectors. Calculated as the L1 dispersion of the score on each component divided by the L1 dispersion in the original data.
`projPoints`	the matrix of projected points in terms of the original coordinates. The matrix has dimension nrow(X) x ncol(X).
`minobjectives`	the L1 distance of points to their projections in the fitted subspace.

References

Brooks J.P. and Dula J.H. (2017) Estimating L1-Norm Best-Fit Lines, submitted.

Examples

```
##for a 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100) +
                matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
mysharpel1pca <- sharpel1pca(X)

##projects data into 2 dimensions.
mysharpel1pca <- sharpel1pca(X, projDim=2, center=FALSE, projections="l1")

## plot first two scores
plot(mysharpel1pca$scores)
```

---

sharpel1rs *SharpEl1-RS*

Description

Fits a line in the presence of missing data based on an L1-norm criterion.

Usage

```
sharpel1rs(X, projDim=1, center=TRUE, projections="none")
```

Arguments

X	data, must be in matrix or table form.
projDim	number of dimensions to project data into, must be an integer, default is 1.
center	whether to center the data using the median, default is TRUE.
projections	whether to calculate reconstructions and scores using the L1 norm ("l1") the L2 norm ("l2") or not at all ("none", default).

Details

The algorithm finds successive, orthogonal fitted lines in the data.

Value

'sharpel1rs' returns a list with class "sharpel1rs" containing the following components:

loadings	the matrix of variable loadings. The matrix has dimension ncol(X) x projDim. The columns define the projected subspace.
scores	the matrix of projected points. The matrix has dimension nrow(X) x projDim.
dispExp	the proportion of L1 dispersion explained by the loadings vectors. Calculated as the L1 dispersion of the score on each component divided by the L1 dispersion in the original data.

`projPoints` the matrix of projected points in terms of the original coordinates. The matrix has dimension nrow(X) x ncol(X).

`minobjectives` the L1 distance of points to their projections in the fitted subspace.

References

Valizadeh Gamchi, F. and Brooks J.P. (2023), working paper.

Examples

```
##for a 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100) +
                matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
mysharpel1rs <- sharpel1rs(X)

##projects data into 2 dimensions.
mysharpel1rs <- sharpel1rs(X, projDim=2, center=FALSE, projections="l1")

## plot first two scores
plot(mysharpel1rs$scores)
```

---

sparsel1pca *SparsEl1-PCA*

Description

L1-norm line fitting with L1-regularization.

Usage

```
sparsel1pca(X, projDim=1, center=TRUE, projections="none", lambda=0)
```

Arguments

`X` data, must be in `matrix` or table form.

`projDim` number of dimensions to project data into, must be an integer, default is 1.

`center` whether to center the data using the median, default is TRUE.

`projections` whether to calculate reconstructions and scores using the L1 norm ("l1") the L2 norm ("l2") or not at all ("none", default).

`lambda` If negative and number of rows is at most 100, calculates all possible breakpoints for the regularization parameter. Otherwise, fits a regularlized line with lambda set to that value.

Details

The calculation is performed according to the algorithm described by Ling and Brooks (2023, working paper). The algorithm finds successive, orthogonal fitted lines in the data.

Value

'sparsel1pca' returns a list with class "sparsel1pca" containing the following components:

`loadings`	the matrix of variable loadings. The matrix has dimension ncol(X) x projDim. The columns define the projected subspace.
`scores`	the matrix of projected points. The matrix has dimension nrow(X) x projDim.
`dispExp`	the proportion of L1 dispersion explained by the loadings vectors. Calculated as the L1 dispersion of the score on each component divided by the L1 dispersion in the original data.
`projPoints`	the matrix of projected points in terms of the original coordinates. The matrix has dimension nrow(X) x ncol(X).
`minobjectives`	the L1 distance of points to their projections in the fitted subspace.

References

Ling, X. and Brooks J.P. (2023) L1-Norm Regularized L1-Norm Best-Fit Lines, working paper.

Examples

```
##for a 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100) +
                matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
mysparsel1pca <- sparsel1pca(X, lambda=0.5)

##projects data into 2 dimensions.
mysparsel1pca <- sparsel1pca(X, projDim=2, center=FALSE, projections="l1", lambda=0.5)

## plot first two scores
plot(mysparsel1pca$scores)
```

`weightedL1Distance` *Weighted L1 Distance*

Description

Provides the (weighted) L1-norm distances and total distance of points to a subspace.

Usage

```
weightedL1Distance(X, loadings, weights)
```

Arguments

`X`	data, in `matrix` or table form
`loadings`	an orthonormal matrix of loadings vectors
`weights`	a list of weights for loadings vectors

Details

The reconstructions are calculated by solving a linear program. Then the weights are applied to the distances.

Value

'weightedL1Distance' returns a list containing the following components:

`wDistances`	list of weighted distances
`totalDistance`	total distance

`wl1pca` *wPCA*

Description

Performs a principal component analysis using the algorithm wPCA described by Park and Klabjan (2016).

Usage

```
wl1pca(X, projDim=1, center=TRUE, projections="l2",
         tolerance=0.001, iterations=200, beta=0.99)
```

Arguments

`X`	data, must be in `matrix` or table form.
`projDim`	number of dimensions to project data into, must be an integer, default is 1.
`center`	whether to center the data using the mean, default is TRUE
`projections`	whether to calculate projections (reconstructions and scores) using the L2 norm ("l2", default) or the L1 norm ("l1").
`tolerance`	for testing convergence; if the sum of absolute values of loadings vectors is smaller, then the algorithm terminates.
`iterations`	maximum number of iterations in optimization routine.
`beta`	algorithm parameter to set up bound for weights.

Details

The calculation is performed according to the algorithm described by Park and Klabjan (2016). The method is an iteratively reweighted least squares algorithm for L1-norm principal component analysis.

Value

'wl1pca' returns a list with class "wl1pca" containing the following components:

`loadings`	the matrix of variable loadings. The matrix has dimension ncol(X) x projDim. The columns define the projected subspace.
`scores`	the matrix of projected points. The matrix has dimension nrow(X) x projDim.
`projPoints`	the matrix of L2 projections points on the fitted subspace in terms of the original coordinates. The matrix has dimension nrow(X) x ncol(X).
`L1error`	sum of the L1 norm of reconstruction errors.
`nIter`	number of iterations.
`ElapsedTime`	elapsed time.

References

Park, Y.W. and Klabjan, D. (2016) Iteratively Reweighted Least Squares Algorithms for L1-Norm Principal Component Analysis, *IEEE International Conference on Data Mining (ICDM)*, 2016. DOI: 10.1109/ICDM.2016.0054

Examples

```
##for 100x10 data matrix X,
## lying (mostly) in the subspace defined by the first 2 unit vectors,
## projects data into 1 dimension.
X <- matrix(c(runif(100*2, -10, 10), rep(0,100*8)),nrow=100) +
              matrix(c(rep(0,100*2),rnorm(100*8,0,0.1)),ncol=10)
mywl1pca <- wl1pca(X)

##projects data into 2 dimensions.
mywl1pca <- wl1pca(X, projDim=2, center=FALSE)

## plot first two scores
plot(mywl1pca$scores)
```

Index