Package ‘pcaL1’

July 21, 2017

Version 1.5.2
Date 2017-07-20
Title L1-Norm PCA Methods
License GPL (>= 3)
Author Sapan Jot <sapan.madaan@gmail.com>, Paul Brooks
 ypsum Nmadaan@gmailcom>, Andrea Visentin <andrea.visentin@insight-centre.org>,
  Young Woong Park <ywpark@mail.smu.edu>,
  and Yi-Hui Zhou <yihui_zhou@ncsu.edu>
Maintainer Paul Brooks <jpbrooks@vcu.edu>
Description Implementations of several methods for principal component analysis
  using the L1 norm. The package depends on COIN-OR Clp version >=
  1.12.0. 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).
URL http://www.optimization-online.org/DB_HTML/2012/04/3436.html,
  http://www.coin-or.org
SystemRequirements COIN-OR Clp (>= 1.12.0)
NeedsCompilation yes
Repository CRAN
Date/Publication 2017-07-20 22:59:46 UTC
This package contains implementations of six principal component analysis methods using the L1 norm. The package depends on COIN-OR Clp version >= 1.12.0. 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).
pcaL1-package

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 SharpeL1-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@mail.smu.edu>, and Yi-Hui Zhou <yihui_zhou@ncsu.edu>
Maintainer: Paul Brooks <jpbrooks@vcu.edu>

References

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


Examples

```r
# 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

```r
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

'1l1pca' 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 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


Examples

```r
# 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)
```

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 Re-
verse Iterative Linear Programming, Joint European Conference on Machine Learning and Knowl-
edge Discovery in Databases, 593-605. DOI:10.1007/978-3-319-46227-1_37

Examples

```r
## 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) +
```
l1pcahp <- 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**

```r
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).
11projection

References


Examples

```r
# 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)
```

11projection    \hspace{1cm} \textit{L1 Projection}

Description

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

Usage

```r
11projection(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

'11projection' 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)
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


See Also

awl1pca
**l2projection**

**L2 Projection**

**Description**

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

**Usage**

```r
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**

```r
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


Examples

```r
# 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)
```
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 = 0.000000001, 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


Examples

```r
## 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="11")

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

---

### plot.awl1pca

#### Plot an awl1pca Object

**Description**

Plots the scores on the first two principal components.

**Usage**

```r
## 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.
plot.l1pca

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

```r
# 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

```r
## 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

```r
# 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 <- l1pcah(X)

# projects data into 2 dimensions.
```
**plot.l1pcastar**

```r
myl1pcahp <- l1pcahp(X, projDim=2, center=FALSE, projections="11")

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

---

### plot.l1pcastar

**Plot an L1pcastar Object**

**Description**

Plots the scores on the first two principal components.

**Usage**

```r
## 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**

```r
##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="11")

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

Plots the scores on the first two principal components.

Usage

```r
## 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

```r
## 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(rnorm(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

```r
## 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

```r
## 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)
```
Description

Plots the scores on the first two principal components.

Usage

```r
## 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

```r
## 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

'sharpe1l1pca' returns a list with class "sharpe1l1pca" 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

Examples

```r
# 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)
mysharpe1pca <- sharpe1pca(X)

# projects data into 2 dimensions.
mysharpe1pca <- sharpe1pca(X, projDim=2, center=FALSE, projections="l1")

# plot first two scores
plot(mysharpe1pca$scores)
```

Description

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

Usage

```r
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


Examples

```r
## 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

*Topic multivariate
  l1pca, 5
  l1pcastar, 8
  plot.awl1pca, 14
  plot.l1pca, 15
  plot.l1pcahp, 16
  plot.l1pcastar, 17
  plot.pcal1, 18
  plot.pcalp, 19
  plot.sharpel1pca, 20
  plot.wl1pca, 21
  sharpel1pca, 22

*Topic package
  pcal1-package, 2

awl1pca, 4, 10
l1pca, 5, 15
l1pcahp, 6
l1pcastar, 8, 15–17, 21
l1projections, 9
L2PCA_approx, 10
l2pcaapprox (L2PCA_approx), 10
l2projection, 11

pcal1 (pcal1-package), 2
pcal1, 11, 18
pcal1-package, 2
pcalp, 13, 19
plot.awl1pca, 14
plot.l1pca, 15
plot.l1pcahp, 16
plot.l1pcastar, 17
plot.pcal1, 18
plot.pcalp, 19
plot.sharpel1pca, 20
plot.wl1pca, 21

sharpenl1pca, 20, 22

wl1pca, 23