Package ‘idm’

August 25, 2015

**Type** Package

**Title** Incremental Decomposition Methods

**Version** 1.2

**Date** 2015-08-25

**Author** Alfonso Iodice D’Enza [aut], Angelos Markos [aut, cre], Davide Buttarazzi [ctb]

**Maintainer** Angelos Markos <amarkos@gmail.com>

**Depends** corpcor, ggplot2, dummies, animation, ca

**SystemRequirements** ImageMagick (http://imagemagick.org) or GraphicsMagick (http://www.graphicsmagick.org)

**Description** Principal Component Analysis and Multiple Correspondence Analysis using incremental eigenvalue decomposition methods.

**License** GPL (>= 2)

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2015-08-25 01:13:02

**R topics documented:**

idm-package .................................................. 2
add_eig ...................................................... 2
add_svd ...................................................... 4
do_eig ......................................................... 6
i_mca .......................................................... 7
i_pca .......................................................... 9
plot_i_mca ................................................... 11
plot_i_pca ................................................... 13
transform_z .................................................. 14
tweet .......................................................... 15
update_mca ................................................... 16
update_pca ................................................... 18
women ......................................................... 19

Index 21
Package: idm
Type: Package
Version: 1.1
Date: 2015-08-24
License: GPL (>=2)

Author(s)
Alfonso Iodice D’ Enza [aut], Angelos Markos [aut, cre], Davide Buttarazzi [ctb]

References


Description
Adds two eigenspaces using incremental eigendecomposition

Usage
add_eig(m1, m2)
add_eig

Arguments

m1 A list describing the eigenspace of a data matrix, with components
u Left eigenvectors
v Right eigenvectors
m Number of cases
d Eigenvalues
orgn Data mean

m2 A list describing the eigenspace of a data matrix, with components
u Left eigenvectors
v Right eigenvectors
m Number of cases
d Eigenvalues
orgn Data mean

Value

A list describing the eigenspace of a data matrix, with components
u Left eigenvectors
v Right eigenvectors
m Number of cases
d Eigenvalues
orgn Data mean

References


See Also

add_svd, update_pca, i_pca, i_mca, update_mca, do_eig

Examples

#Iris species
data("iris", package = "datasets")
X = iris[,,-5]
#obtain two eigenspaces
m1 = do_eig(X[1:50,])
m2 = do_eig(X[51:150,])
#add the two eigenspaces keeping track of the data mean
m12 = add_eig(m1, m2)
#similar to the SVD of the mean-centered X
add_svd

Adds two eigenspaces using incremental SVD (with or without mean update)

Description

This function adds the eigenspace of an incoming data block to an existing eigenspace using the incremental SVD method described by Zha & Simon (1999), Levy and Lindenbaum (2000), Brand (2002) and Baker (2012). New data blocks are added row-wise. The procedure can optionally keep track of the data mean using the orgn argument, as described in Ross et al. (2008) and Iodice D’Enza & Markos (2015).

Usage

add_svd(eg, B, m, current_rank, orgn, ff = 0)

Arguments

dep A list describing the SVD of a data matrix, with components
  u Left singular vectors
  v Right singular vectors
  d Singular values

B Incoming data block (rows > columns)

m Number of cases (rows) processed up to this point

current_rank Rank of approximation; if empty then full rank is used

orgn Data mean; if empty then data is assumed as mean-centered

ff Number between 0 and 1 indicating the forgetting factor used to down-weight the contribution of earlier data blocks to the current solution. When ff = 0 (default) no forgetting occurs

Value

A list describing the SVD of a data matrix, with components

u Left singular vectors
d Singular values
v Right singular vectors
m Number of cases
orgn Data mean; returned only if ‘orgn’ is given as input
References


See Also

`add_eig, update_mca, i_pca, i_mca`

Examples

```r
# Example 1 - Update SVD (full rank)
data("iris", package = "datasets")
# Obtain the SVD of the first 50 Iris species
X = iris[, -5]
eg = svd(X[1:50,])
# Update the eigenspace of the remaining species to
eg = add_svd(eg, data.matrix(X[c(51:150),]), 50)
# Similar results to svd(X)

# Example 2 - Update SVD with mean update (full rank)
data("iris", package = "datasets")
# Obtain the eigenspace of the first 50 Iris species
X = iris[, -5]
eg = do_eig(X[1:50,])
# Update the eigenspace of the remaining species to
eg = add_svd(eg, data.matrix(X[c(51:150),]), 50, eg$orgn)
# eg$sv corresponds to PCA loadings on the covariance matrix of X

# Example 3 - Update SVD (low-rank approximation: current_rank = 2)
data("iris", package = "datasets")
# Obtain the SVD of the first 50 Iris species
X = iris[, -5]
eg = svd(X[1:50,])
# Update the eigenspace of the remaining species to
eg = add_svd(eg, data.matrix(X[c(51:150),]), 50, 2)
# Similar results to svd(X)
```
**do_eig**  
*Computes the eigenspace of a data matrix*

**Description**

This function computes the eigenspace of a mean-centered data matrix.

**Usage**

```r
do_eig(data)
```

**Arguments**

- `data`  
a matrix or data frame

**Value**

A list describing the eigenspace of a data matrix, with components

- `vctCol`  
Left eigenvectors
- `vct`  
Right eigenvectors
- `n`  
Number of cases
- `val`  
Eigenvalues
- `orgn`  
Data mean

**See Also**

- `add_svd`
- `update_pca`
- `i_pca`
- `i_mca`
- `do_eig`

**Examples**

```r
data("iris", package = "datasets")
m = do_eig(iris[, -5])
#corresponds to the SVD of the centered data matrix
```
Incremental Multiple Correspondence Analysis (MCA)

Description
This function computes the Multiple Correspondence Analysis (MCA) solution on the indicator matrix using two incremental methods described in Iodice D'Enza & Markos (2015).

Usage
i_mca(data1, data2, method=c("exact","live"), nchunk = 2, current_rank = 2, ff = 0, disk=FALSE)

Arguments
- data1: A numeric matrix or data frame of starting data
- data2: A numeric matrix or data frame of incoming data
- method: String specifying the type of implementation: "exact" or "live". "exact" refers to the case when all the data is available from the start and dimension reduction is based on the method of Hall et al. (2002). "live" refers to the case when new data comes in as data flows and dimension reduction is based on the method of Ross et al. (2008). The main difference between the two approaches lies in the calculation of the column margins of the input matrix. For the "exact" approach, the analysis is based on the "global" margins, that is, the margins of the whole indicator matrix, which is available in advance. For the "live" approach, the whole matrix is unknown and the global margins are approximated by the "local" margins, that is, the average margins of the data analysed so far. A detailed description of the two implementations is provided in Iodice D' Enza & Markos (2015).
- nchunk: Number of incoming data chunks (equal splits of 'data2') or a Vector with the row size of each incoming data chunk
- current_rank: Rank of the approximation or number of retained dimensions (default = 2)
- ff: Number between 0 and 1 indicating the "forgetting factor" used to down-weight the contribution of earlier data blocks to the current solution. When ff = 0 (default) no forgetting occurs; applicable only when method ="live"
- disk: Logical indicating whether the output is saved to hard disk

Value
- colpcoordstart: Column principal coordinates of the starting solution
- colpcoord: Column principal coordinates of the final solution
- rowpcoordstart: Row principal coordinates of the starting data
- rowpcoord: Row principal coordinates of the final solution
- levelnames: Column names
colctr  Column contributions  
colcor  Column squared correlations  
rowctr  Row contributions  
rowcor  Row squared correlations  
sv  Eigenvalues  
rowmass  Row masses  
colmass  Column masses  
inertiaNe  Percentages of explained (adjusted) inertia  
nchunk  Number of incoming data chunks  
disk  Logical indicating whether then output is saved to hard disk  
allrowcoords  A list containing the row scores on the principal components produced after each data chunk is analyzed; applicable only when disk = FALSE  
ff  Number between 0 and 1 indicating the "forgetting factor" used to down-weight the contribution of earlier data blocks to the current solution  
allcolcoords  A list containing the variable loadings on the principal components produced after each data chunk is analyzed; applicable only when disk = FALSE  
allcolctr  A list containing the column contributions after each data chunk is analyzed; applicable only when disk = FALSE  
allcolcor  A list containing the column squared correlations produced after each data chunk is analyzed; applicable only when disk = FALSE  
allrowctr  A list containing the row contributions after each data chunk is analyzed; applicable only when disk = FALSE  
allrowcor  A list containing the row squared correlations produced after each data chunk is analyzed; applicable only when disk = FALSE  
References  
See Also  
*update_mca*, *i_pca*, *update_pca*, *add_svd*, *add_eig*


Examples

```
# Example 1 - Exact case
data("women", package = "idm")
nc = 5 # number of chunks
res_iMCAh = i_mcah(data1 = women[1:300,1:7], data2 = women[301:2107,1:7],
method = "exact", nchunk = nc)
# static MCA plot of attributes on axes 2 and 3
plot(c = res_iMCAh, dim = c(2,3), what = c(FALSE,TRUE), animation = FALSE)

# don't test is used here because the code calls the saveGIF function of the animation package
# which requires ImageMagick or GraphicsMagick to be installed in your system
# See help(im.convert) for details on the configuration of ImageMagick or GraphicsMagick.
# Creates animated GIF movies for objects and variables
plot(res_iMCAh, animation = TRUE, frames = 10)
```

```
# Example 2 - Live case
data("tweet", package = "idm")
nc = 5
# provide attributes with custom labels
# mimics the 'live' MCA implementation
res_iMCA1 = i_mca(data1 = tweet[1:100,], data2 = tweet[101:1000,],
method="live", nchunk = nc, current_rank=2)

# don't test is used here because the code calls the saveGIF function of the animation package
# which requires ImageMagick or GraphicsMagick to be installed in your system
# See help(im.convert) for details on the configuration of ImageMagick or GraphicsMagick.
# Creates animated GIF movies for objects and variables
plot(res_iMCA1, labels = labels, animation = TRUE, frames = 10)
```

---

**i_pca**

**Incremental Principal Component Analysis (PCA)**

**Description**

This function computes the Principal Component Analysis (PCA) solution on the covariance matrix using the incremental method of Hall, Marshall & Martin (2002).

**Usage**

```
i_pca(data1, data2, nchunk = 2, disk = FALSE)
```

**Arguments**

- `data1` Matrix or data frame of starting data
- `data2` Matrix or data frame of incoming data
nchunk Number of incoming data chunks (equal splits of 'data2') or a Vector with the row size of each incoming data chunk

disk Logical indicating whether the output is saved to hard disk

Value

scoreStart Row scores on the principal components of the starting solution
loadStart Variable loadings of the starting solution
colpcoordStart Variable loadings of the starting data
colpcoord Variable loadings of the final solution
rowpcoordStart Row scores on the principal components of the starting data
rowpcoord Row scores on the principal components of the final solution
levelnames Column names
rownames Row names
rowctr Row contributions
rowcor Row squared correlations
sv Eigenvalues
inertia_e Percentage of explained variance
nchunk Number of incoming data chunks
disk Logical indicating whether the output is saved to hard disk
allrowcoords A list containing the row scores on the principal components produced after each data chunk is analyzed; applicable only when disk = FALSE
allcolcoords A list containing the variable loadings on the principal components produced after each data chunk is analyzed; applicable only when disk = FALSE
allrowctr A list containing the row contributions after each data chunk is analyzed; applicable only when disk = FALSE
allrowcor A list containing the row squared correlations produced after each data chunk is analyzed; applicable only when disk = FALSE

References


See Also

update_pca, i_mca, update_mca, add_svd, add_eig
Examples

```r
# Iris Species
data("iris", package = "datasets")
X = iris[, -5]
res_ipCA = i_pca(data1 = X[1:50,], data2 = X[51:150,], nchunk = c(50, 50))
# static plot
plot(res_ipCA, animation = FALSE)

# donttest is used here because the code calls the saveGIF function of the animation package
# which requires ImageMagick or GraphicsMagick to be installed in your system
# See help(im.convert) for details on the configuration of ImageMagick or GraphicsMagick.
# Creates animated GIF movies for objects and variables
plot(res_ipCA, animation = TRUE, frames = 10)
```

```r
# Daily Closing Prices of Major European Stock Indices, 1991-1998
data("EuStockMarkets", package = "datasets")
res_ipCA = i_pca(data1 = EuStockMarkets[1:50,], data2 = EuStockMarkets[51:1860,], nchunk = 5)

# donttest is used here because the code calls the saveGIF function of the animation package
# which requires ImageMagick or GraphicsMagick to be installed in your system
# See help(im.convert) for details on the configuration of ImageMagick or GraphicsMagick.
# Creates animated GIF movies for objects and variables
plot(res_ipCA, animation = TRUE, frames = 10)
```

plot.i_mca

**Plotting 2D maps in Multiple Correspondence Analysis**

Description

Graphical display of Multiple Correspondence Analysis results in two dimensions

Usage

```r
## S3 method for class 'i_mca'
plot(x, dims = c(1, 2), what = c(TRUE, TRUE),
     contrib = "none", dataname = NULL, labels = NULL, animation = TRUE, frames = 10,
     zoom = TRUE, ...)
```

Arguments

- `x` Multiple correspondence analysis object returned by `i_mca`
- `dims` Numerical vector of length 2 indicating the dimensions to plot on horizontal and vertical axes respectively; default is first dimension horizontal and second dimension vertical
- `what` Vector of two logicals specifying the contents of the plot(s). First entry indicates if the rows (objects) are displayed and the second entry if the columns (attributes) are displayed
plot.i_mca

contrib Vector of two character strings specifying if attribute contributions should be represented by different label size. Available options are “none” (contributions are not indicated in the plot) “cor” (relative contributions are indicated by label size) “ctr” (absolute contributions are indicated by label size) The higher the contribution of a point, the larger its label size. Default is ”none”
datename String prefix used for custom naming of output files; default is the name of the output object
labels String vector of variable labels
animation Logical indicating whether animated GIF files are created and saved to the hard drive or a static plot is obtained
frames Number of animation frames shown per iteration; applicable only when animation = TRUE
zoom Logical indicating whether axis limits change during the animation creating a zooming effect; applicable only when animation = TRUE
...
Further arguments passed to plot and points

Details

The function plot.i_mca makes a two-dimensional map of the object created by i_mca with respect to two selected dimensions. In this map both the row and column points are scaled to have inertias (weighted variances) equal to the principal inertia (eigenvalue or squared singular value) along the principal axes, that is both rows and columns are in principal coordinates.

References


See Also

plot.i_pca

Examples

data("women", package="idm")
nc = 4 # number of chunks
### Different approaches to incremental MCA:
# Live case
res_iMCA1 = i_mca(data1 = women[1:50,1:4],data2 = women[51:300,1:4],method = "live",nchunk = nc)
#static plot, final solution
plot(res_iMCA1, contrib="ctr", animation = FALSE)

#donttest is used here because the code calls the saveGIF function of the animation package
#which requires ImageMagick or GraphicsMagick to be installed in your system
plot.i_pca

#See help(im.convert) for details on the configuration of ImageMagick or GraphicsMagick.
#Creates animated GIF movies for objects and variables
plot(res.IMCA1, contrib = "ctr", animation = TRUE, frames = 10)

plot.i_pca

Plotting 2D maps in Principal Component Analysis

Description

Graphical display of Principal Component Analysis results in two dimensions

Usage

## S3 method for class 'i_pca'
plot(x, dims = c(1,2), what = c(TRUE,TRUE),
     dataname = NULL, labels = NULL, animation = TRUE, frames = 10, zoom = TRUE, ...)

Arguments

x Principal component analysis object returned by i_pca
dims Numerical vector of length 2 indicating the dimensions to plot on horizontal and vertical axes respectively; default is first dimension horizontal and second dimension vertical
what Vector of two logicals specifying the contents of the plot(s). First entry indicates if the rows (objects) are displayed and the second entry if the columns (attributes) are displayed
dataname String prefix used for custom naming of output files; default is the name of the output object
labels String vector of variable labels
animation Logical indicating whether animated GIF files are created and saved to the hard drive or a static plot is obtained
frames Number of animation frames shown per iteration; applicable only when animation = TRUE
zoom Logical indicating whether axis limits change during the animation creating a zooming effect; applicable only when animation = TRUE
...
Further arguments passed to plot and points

Details

The function plot.i_pca makes a two-dimensional map of the object created by i_pca with respect to two selected dimensions.
transform_z

Computes indicator and standardized residual matrices

Description

This function returns the indicator (or dummy) matrix and the standardized residual matrix of a given data matrix.

Usage

transform_z(data, is.weight = TRUE, is.exact = TRUE, r = 1, c = 1)

Arguments

data Matrix or data frame

is.weight Logical indicating whether computations are based on user-defined weights/masses

is.exact Logical indicating whether the computations mimic the case when 'exact' or 'live' MCA is applied

r Row masses; applicable only when is.weight = TRUE

c Column masses; applicable only when is.weight = TRUE
Value

\begin{itemize}
\item \( dZ \): Indicator matrix
\item \( Q \): Number of variables
\item \( J \): Number of variable attributes
\item \( r \): Row masses
\item \( c \): Column masses
\item \( SZ \): Standardized residual matrix
\end{itemize}

Examples

```r
data(women, package="idm")
outZ = transform_z(women[,c(1:7)],is.weight=FALSE)
```

Description

The data set refers to a small corpus of messages or tweets mentioning seven major hotel brands. It was gathered by continuously querying and archiving the Twitter Streaming API service, using the `twitter` package in \( R \). A total of 7,296 tweets were extracted within a time period of 6 days, from June 23th to June 28th 2013. Only tweets in the English language were considered. A sentiment polarity variable was calculated, indicating the sentiment value of each message and a third variable, user visibility or popularity, as measured by the number of followers each user had, was also included in the dataset.

Usage

```r
data("tweet")
```

Format

A data frame with the following variables:

- **Brand**:
  \( 1=\text{Hilton}, 2=\text{Intercontinental}, 3=\text{Marriott}, 4=\text{Bestwestern}, 5=\text{Starwood}, 6=\text{Hyatt}, 7=\text{Choice} \)

- **Sentiment**:
  \( 1=\text{negative (-), 2=mixed (+/-), 3=positive (+), 4=very positive (++)} \)

- **UserVis**:
  User popularity/visibility in Twitter: \( 1=\text{low}, 2=\text{medium}, 3=\text{high} \)

References


Examples

```r
data(tweet)
```
**update_mca**  
*Updates a Multiple Correspondence Analysis solution*

**Description**

This function updates the Multiple Correspondence Analysis (MCA) solution on the indicator matrix using the incremental method of Ross et al. (2008).

**Usage**

```r
update_mca(mca, incdata, current_rank = 2, ff = 0)
```

**Arguments**

- `incdata`  
  Matrix of incoming data

- `mca`  
  A list describing an MCA solution with components  
  - `indmat` Indicator matrix  
  - `colcoord` Column standard coordinates  
  - `rowcoord` Row standard coordinates  
  - `colpcoord` Column principal coordinates  
  - `rowpcoord` Row principal coordinates  
  - `colctr` Column contributions  
  - `colcor` Column squared correlations  
  - `rowctr` Row contributions  
  - `rowcor` Row squared correlations  
  - `sv` Eigenvalues  
  - `rowmass` Row masses  
  - `colmass` Column masses  
  - `levelnames` Attribute names  
  - `m` Number of cases processed up to this point  
  - `orgn` Data mean  
  - `inertiaNe` Percentages of explained (adjusted) inertia  

- `current_rank`  
  Rank of the approximation or number of retained dimensions; default = 2

- `ff`  
  Number between 0 and 1 indicating the "forgetting factor" used to down-weight the contribution of earlier data blocks to the current solution. When f = 0 (default) no forgetting occurs

**Value**

A list describing an MCA solution with components

- `indmat` Indicator matrix
- `colcoord` Column standard coordinates
- `rowcoord` Row standard coordinates
- `colpcoord` Column principal coordinates
update_mca

rowpcoord  Row principal coordinates
colctr    Column contributions
colcor    Column squared correlations
rowctr    Row contributions
rowcor    Row squared correlations
sv         Eigenvalues
rowmass    Row masses
colmass    Column masses
levelnames Attribute names
m          Number of cases processed up to this point
orgn       Data mean
inertia_e  Percentages of explained (adjusted) inertia
ff         Number between 0 and 1 indicating the "forgetting factor"

References


See Also

update_pca, i_pca, add_svd, add_eig

Examples

data(women, package="idm")
dat = women[,c(1:4)]
n0 = 301
outmca = mjca(dat[1:n0,],lambda="indicator",ret=TRUE)
outZ = transform_z(dat[1:n0,],is.weight=FALSE)
outmca$m = n0
outmca$rowmass = outZ$r
outmca$orgn = colMeans(outZ$SZ[1:n0,])
nchunk = seq(n0,2107,258)
for (k in c(1:(length(nchunk)-1))){
    outmca = update_mca(outmca,dat[c((nchunk[k]+1):nchunk[k+1]),])
}
#to get coordinates on a similar scale to the ones produced with batch mca
outmca$colpcoord = outmca$colpcoord/sqrt(length(nchunk))
outmca$rowpcoord = outmca$rowpcoord/sqrt(length(nchunk))

plot(outmca, what=c(FALSE, TRUE), animation = FALSE)
**update_pca**

*Updates a Principal Component Analysis solution*

**Description**

This function updates the Principal Component Analysis (PCA) solution on the covariance matrix using the incremental method of Hall, Marshall & Martin (2002)

**Usage**

`update_pca(eg, incdata)`

**Arguments**

- **incdata**
  - Matrix of incoming data
- **eg**
  - A list describing the eigenspace of a data matrix, with components
    - `u` Left eigenvectors
    - `v` Right eigenvectors
    - `m` Number of cases
    - `d` Eigenvalues
    - `orgn` Data mean

**Value**

A list describing the eigenspace of a data matrix and a PCA solution, with components

- `u` Left eigenvectors
- `v` Right eigenvectors
- `m` Number of cases
- `d` Eigenvalues
- `orgn` Data mean
- `colpcoord` Variable loadings
- `rowpcoord` Row scores on the principal components
- `levelname` Column names
- `rowctr` Row contributions
- `rowcor` Row squared correlations
- `sv` Standard deviations of the principal components
- `inertia_e` Percentages of explained variance

**References**


women

See Also
update_mca, i_pca, i_mca, add_svd, add_eig

Examples

data(iris, package="datasets")
dat = iris[, -5]
eg = do_eig(dat[1:50, -5])
aa = seq(50, 150, 50)
for (k in c(1:2)) {
  eg = update_pca(eg, dat[c(aa[k]+1):aa[k+1]], )
}
plot(eg, animation = FALSE)

women

women data set

Description

The data are from the third Family and Changing Gender Roles survey conducted in 2002. The questions retained are those related to working women in Spain and the effect on the family. A total of 2,107 respondents answered eight questions on a 5-point Likert scale, as well as four demographic variables (gender, marital status, education and age). There are no cases with missing data.

Usage

data("women")

Format

A data frame with the following variables:

A  "a working mother can establish a warm relationship with her child"
   1=strongly agree, 2=agree, 3=neither agree or disagree, 4=disagree, 5=strongly disagree

B  "a pre-school child suffers if his or her mother works"
   1=strongly agree, 2=agree, 3=neither agree or disagree, 4=disagree, 5=strongly disagree

C  "when a woman works the family life suffers"
   1=strongly agree, 2=agree, 3=neither agree or disagree, 4=disagree, 5=strongly disagree

D  "what women really want is a home and kids"
   1=strongly agree, 2=agree, 3=neither agree or disagree, 4=disagree, 5=strongly agree
E "running a household is just as satisfying as a paid job"
   1=strongly agree, 2=agree, 3=neither agree or disagree, 4=disagree, 5=strongly disagree
F "work is best for a woman’s independence"
   1=strongly agree, 2=agree, 3=neither agree or disagree, 4=disagree, 5=strongly disagree
G "a man’s job is to work; a woman’s job is the household"
   1=strongly agree, 2=agree, 3=neither agree or disagree, 4=disagree, 5=strongly disagree
H "working women should get paid maternity leave"
   1=strongly agree, 2=agree, 3=neither agree or disagree, 4=disagree, 5=strongly disagree

g gender: 1=male, 2=female
m marital status: 1=married/living as married, 2=widowed, 3=divorced, 4=separated, but married, 5=single, never married
e education: 1=no formal education, 2=lowest education, 3=above lowest education, 4=highest secondary completed, 5=above higher secondary level, below full university, 6=university degree completed
   a age: 1=16-25 years, 2=26-35, 3=36-45, 4=46-55, 5=56-65, 6=66 and older

Source


References


Examples

data(women)
Index

*Topic datasets
  tweet, 15
  women, 19
*Topic package
  idm-package, 2

add_eig, 2, 5, 8, 10, 17, 19
add_svd, 3, 4, 6, 8, 10, 17, 19

do_eig, 3, 6, 6
i_mca, 3, 5, 6, 7, 10, 11, 19
i_pca, 3, 5, 6, 8, 9, 13, 17, 19
idm (idm-package), 2
idm-package, 2

plot, 12, 13
plot.i_mca, 11, 14
plot.i_pca, 12, 13
points, 12, 13

transform_z, 14
tweet, 15

update_mca, 3, 5, 8, 10, 16, 19
update_pca, 3, 6, 8, 10, 17, 18

women, 19