# Package ‘idm’

August 30, 2018

**Type** Package

**Title** Incremental Decomposition Methods

**Version** 1.8.2

**Date** 2018-08-30

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

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

**Depends** R (>= 2.10), ggplot2, animation, dummies

**Imports** corpcor, ca, ggrepel

**Suggests** caret

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

**Description** Incremental Multiple Correspondence Analysis and Principal Component Analysis.

**License** GPL (>= 2)

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2018-08-30 08:10:15 UTC

## R topics documented:

`idm-package` *Incremental Decomposition Methods*

## Description

Incremental Multiple Correspondence Analysis and Principal Component Analysis

## Details

| | |
|---|---|
| Package: | idm |
| Type: | Package |
| Version: | 1.8.2 |
| Date: | 2018-08-30 |
| License: | GPL (>=2) |

## Author(s)

Alfonso Iodice D' Enza [aut], Angelos Markos [aut, cre], Davide Buttarazzi [ctb]

## References

Hall, P., Marshall, D., & Martin, R. (2002). Adding and subtracting eigenspaces with eigenvalue decomposition and singular value decomposition. *Image and Vision Computing*, *20*(13), 1009-1016.

Iodice D' Enza, A., & Markos, A. (2015). Low-dimensional tracking of association structures in categorical data, *Statistics and Computing*, *25*(5), 1009–1022.

Iodice D'Enza, A., Markos, A., & Buttarazzi, D. (2018). The idm Package: Incremental Decomposition Methods in R. *Journal of Statistical Software, Code Snippets*, *86*(4), 1–24. DOI: 10.18637/jss.v086.c04.

Ross, D. A., Lim, J., Lin, R. S., & Yang, M. H. (2008). Incremental learning for robust visual tracking. *International Journal of Computer Vision*, *77*(1-3), 125–141.

`add_es` *Adds two eigenspaces using block-wise incremental SVD (with or without mean update)*

### Description

This function implements two procedures for updating existing decomposition. When `method="esm"` it adds two eigenspaces using the incremental method of Hall, Marshall & Martin (2002). The results correspond to the eigenspace of the mean-centered and concatenated data. When `method = "isvd"` it adds the eigenspace of an incoming data block to an existing eigenspace using the block-wise incremental singular value decomposition (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_es(eg, eg2, current_rank, ff = 0, method = c("esm", "isvd"))
```

### Arguments

| | |
|---|---|
| `eg` | A list describing the eigenspace of a data matrix, with components<br>`u` Left eigenvectors<br>`v` Right eigenvectors<br>`m` Number of cases<br>`d` Eigenvalues<br>`orgn` Data mean |
| `method` | refers to the procedure being implemented: `"esm"` refers to the eigenspace merge (Hall et al., 2002); `"isvd"` refers to the incremental SVD method, with or without keeping track of the data mean. |
| `eg2` | (*)A list describing the eigenspace of a data matrix, with components<br>`u` Left eigenvectors<br>`v` Right eigenvectors<br>`m` Number of cases<br>`d` Eigenvalues<br>`orgn` Data mean |
| `current_rank` | Rank of approximation; if empty, the full rank is used |
| `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 (*) for `method = "esm"` only; (**) for `method = "isvd"` only. |

### 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


Zha, H., & Simon, H. D. (1999). On updating problems in latent semantic indexing. *SIAM Journal on Scientific Computing*, *21*(2), 782-791.

Levy, A., & Lindenbaum, M. (2000). Sequential Karhunen-Loeve basis extraction and its application to images. *IEEE Transactions on Image Processing*, *9*(8), 1371-1374.

Brand, M. (2002). Incremental singular value decomposition of uncertain data with missing values. In *Computer Vision-ECCV 2002* (pp. 707-720). Springer Berlin Heidelberg.

Ross, D. A., Lim, J., Lin, R. S., & Yang, M. H. (2008). Incremental learning for robust visual tracking. *International Journal of Computer Vision*, *77*(1-3), 125-141.

Baker, C. G., Gallivan, K. A., & Van Dooren, P. (2012). Low-rank incremental methods for computing dominant singular subspaces. *Linear Algebra and its Applications*, *436*(8), 2866-2888.

Iodice D' Enza, A., & Markos, A. (2015). Low-dimensional tracking of association structures in categorical data, *Statistics and Computing*, *25*(5), 1009-1022. Iodice D'Enza, A., Markos, A., & Buttarazzi, D. (2018). The idm Package: Incremental Decomposition Methods in R. *Journal of Statistical Software, Code Snippets*, *86*(4), 1–24. DOI: 10.18637/jss.v086.c04.


### See Also

do_es, i_pca, i_mca, update.i_pca, update.i_mca

### Examples

```
## Example 1 - eigenspace merge (Hall et al., 2002)
#Iris species
data("iris", package = "datasets")
X = iris[,-5]
#obtain two eigenspaces
eg = do_es(X[1:50, ])
eg2 = do_es(X[c(51:150), ])
#add the two eigenspaces keeping track of the data mean
eg12 = add_es(method = "esm", eg, eg2)
#equivalent to the SVD of the mean-centered data (svd(scale(X, center = TRUE,scale = FALSE)))

## Example 2 - block-wise incremental SVD with mean update, full rank (Ross et al., 2008)
data("iris", package = "datasets")
# obtain the eigenspace of the first 50 Iris species
X = iris[,-5]
eg = do_es(X[1:50, ])
#update the eigenspace of the remaining species to
eg_new = add_es(method = "isvd", eg, data.matrix(X[c(51:150), ]))
```

```
#equivalent to the SVD of the mean-centered data (svd(scale(X, center = TRUE, scale = FALSE)))

##Example 3 - incremental SVD with mean update, 2d approximation (Ross et al., 2008)
data("iris", package = "datasets")
# obtain the eigenspace of the first 50 Iris species
X = iris[,-5]
eg = do_es(X[1:50, ])
#update the eigenspace of the remaining species to
eg = add_es(method = "isvd", eg, data.matrix(X[c(51:150), ]),current_rank = 2)
#similar to PCA on the covariance matrix of X (SVD of the mean-centered data)
```

---

do_es *Computes the eigenspace of a data matrix*

---

## Description

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

## Usage

```
do_es(data)
```

## Arguments

| | |
|---|---|
| data | a matrix or data frame |

## 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 |
| smfq | ... |

## See Also

add_es, update.i_pca, i_pca

## Examples

```
#Iris species
data("iris", package = "datasets")
eg = do_es(iris[,-5])
#corresponds to the SVD of the centered data matrix
```

## enron — *enron data set*

### Description

The data set is a subset of the Enron e-mail corpus from the UCI Machine Learning Repository (Lichman, 2013). The original data is a collection of 39,861 email messages with roughly 6 million tokens and a 28,102 term vocabulary. The subset is a binary (presence/absence) data set containing the 80 most frequent words which appear in the original corpus.

### Usage

```
data("enron")
```

### Format

A binary data frame with 39,861 observations (e-mail messages) on 80 variables (words).

### References

Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

### Examples

```
data(enron)
```

## i_mca — *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"), current_rank, nchunk = 2,
 ff = 0, disk = FALSE)
```

### Arguments

| | |
|---|---|
| `data1` | Matrix or data frame of starting data or full data if data2 = NULL |
| `data2` | 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 insofar. A detailed description of the two implementations is provided in Iodice D' Enza & Markos (2015). |
| `current_rank` | Rank of approximation or number of components to compute; if empty, the full rank is used |
| `nchunk` | Number of incoming data chunks (equal splits of 'data2', `default = 2`) or a Vector with the row size of each incoming data chunk |
| `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 then output is saved to hard disk |

### Value

| | |
|---|---|
| `rowpcoord` | Row principal coordinates |
| `colpcoord` | Column principal coordinates |
| `rowcoord` | Row standard coordinates |
| `colcoord` | Column standard coordinates |
| `sv` | Singular values |
| `inertia.e` | Percentages of explained inertia |
| `levelnames` | Column labels |
| `rowctr` | Row contributions |
| `colctr` | Column contributions |
| `rowcor` | Row squared correlations |
| `colcor` | Column squared correlations |
| `rowmass` | Row masses |
| `colmass` | Column masses |
| `nchunk` | A copy of `nchunk` in the return object |
| `disk` | A copy of `disk` in the return object |
| `ff` | A copy of `ff` in the return object |

`allrowcoord` A list containing the row principal coordinates produced after each data chunk is analyzed; returned only when `disk = FALSE`

`allcolcoord` A list containing the column principal coordinates on the principal components produced after each data chunk is analyzed; returned only when `disk = FALSE`

`allrowctr` A list containing the row contributions after each data chunk is analyzed; returned only when `disk = FALSE`

`allcolctr` A list containing the column contributions after each data chunk is analyzed; returned only when `disk = FALSE`

`allrowcor` A list containing the row squared correlations produced after each data chunk is analyzed; returned only when `disk = FALSE`

`allcolcor` A list containing the column squared correlations produced after each data chunk is analyzed; returned only when `disk = FALSE`

## References

Hall, P., Marshall, D., & Martin, R. (2002). Adding and subtracting eigenspaces with eigenvalue decomposition and singular value decomposition. *Image and Vision Computing*, *20*(13), 1009-1016.

Iodice D' Enza, A., & Markos, A. (2015). Low-dimensional tracking of association structures in categorical data, *Statistics and Computing*, *25*(5), 1009–1022.

Iodice D'Enza, A., Markos, A., & Buttarazzi, D. (2018). The idm Package: Incremental Decomposition Methods in R. *Journal of Statistical Software, Code Snippets*, *86*(4), 1–24. DOI: 10.18637/jss.v086.c04.

Ross, D. A., Lim, J., Lin, R. S., & Yang, M. H. (2008). Incremental learning for robust visual tracking. *International Journal of Computer Vision*, *77*(1-3), 125–141.

## See Also

`update.i_mca`, `i_pca`, `update.i_pca`, `add_es`

## Examples

```
##Example 1 - Exact case
data("women", package = "idm")
nc = 5 # number of chunks
res_iMCAh = i_mca(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(x = res_iMCAh, dim = c(2,3), what = c(FALSE,TRUE), animation = FALSE)

#\donttest is used here because the code calls the saveLatex function of the animation package
#which requires ImageMagick or GraphicsMagick and
#Adobe Acrobat Reader to be installed in your system
#Creates animated plot in PDF for objects and variables
plot(res_iMCAh, animation = TRUE, frames = 10, movie_format = 'pdf')
```

```
##Example 2 - Live case
data("tweet", package = "idm")
nc = 5
#provide attributes with custom labels
labels = c("HLTN", "ICN", "MRT","BWN","SWD","HYT","CH", "-", "-/+", "+", "++", "Low", "Med","High")
#mimics the 'live' MCA implementation
res_iMCAl = i_mca(data1 = tweet[1:100,], data2 = tweet[101:1000,],
method="live", nchunk = nc, current_rank = 2)


#\donttest is used here because the code calls the saveLatex function of the animation package
#which requires ImageMagick or GraphicsMagick and
#Adobe Acrobat Reader to be installed in your system
#See help(im.convert) for details on the configuration of ImageMagick or GraphicsMagick.
#Creates animated plot in PDF for observations and variables
plot(res_iMCAl, labels = labels, animation = TRUE, frames = 10, movie_format = 'pdf')
```

---

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, current_rank, nchunk = 2, disk = FALSE)
```

### Arguments

| | |
|---|---|
| `data1` | Matrix or data frame of starting data, or full data if data2 = NULL |
| `data2` | Matrix or data frame of incoming data; omitted when full data is given in data1 |
| `current_rank` | Rank of approximation or number of components to compute; if empty, the full rank is used |
| `nchunk` | Number of incoming data chunks (equal splits of 'data2', `default = 2`) or a Vector with the row size of each incoming data chunk |
| `disk` | Logical indicating whether then output is saved to hard disk |

### Value

| | |
|---|---|
| `rowpcoord` | Row scores on the principal components |
| `colpcoord` | Variable loadings |

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

sv — Singular values

inertia_e — Percentage of explained variance

levelnames — Attribute labels

rowctr — Row contributions

colctr — Column contributions

rowcor — Row squared correlations

colcor — Column squared correlations

nchunk — A copy of nchunk in the return object

disk — A copy of disk in the return object

allrowcoord — A list containing the row scores on the principal components produced after each data chunk is analyzed; returned only when disk = FALSE

allcolcoord — A list containing the variable loadings on the principal components produced after each data chunk is analyzed; returned only when disk = FALSE

allrowctr — A list containing the row contributions after each data chunk is analyzed; returned only when disk = FALSE

allcolctr — A list containing the column contributions after each data chunk is analyzed; returned only when disk = FALSE

allrowcor — A list containing the row squared correlations produced after each data chunk is analyzed; returned only when disk = FALSE

allcolcor — A list containing the column squared correlations produced after each data chunk is analyzed; returned only when disk = FALSE

## References

Hall, P., Marshall, D., & Martin, R. (2002). Adding and subtracting eigenspaces with eigenvalue decomposition and singular value decomposition. *Image and Vision Computing*, *20*(13), 1009-1016.

Iodice D' Enza, A., & Markos, A. (2015). Low-dimensional tracking of association structures in categorical data, *Statistics and Computing*, *25*(5), 1009–1022.

Iodice D'Enza, A., Markos, A., & Buttarazzi, D. (2018). The idm Package: Incremental Decomposition Methods in R. *Journal of Statistical Software, Code Snippets*, *86*(4), 1–24. DOI: 10.18637/jss.v086.c04.

## See Also

update.i_pca, i_mca, update.i_mca, add_es

## Examples

```
data("segmentationData", package = "caret")
#center and standardize variables, keep 58 continuous attributes
HCS = data.frame(scale(segmentationData[,-c(1:3)]))
#abbreviate variable names for plotting
names(HCS) = abbreviate(names(HCS), minlength = 5)
#split the data into starting data and incoming data
data1 = HCS[1:150, ]
data2 = HCS[151:2019, ]
#Incremental PCA on the HCS data set: the incoming data is
#splitted into twenty chunks; the first 5 components/dimensions
#are computed in each update
res_iPCA = i_pca(data1, data2, current_rank = 5, nchunk = 20)
#Static plots
plot(res_iPCA, animation = FALSE)

#\donttest is used here because the code calls the saveLatex function of the animation package
#which requires ImageMagick or GraphicsMagick and
#Adobe Acrobat Reader to be installed in your system
#See help(im.convert) for details on the configuration of ImageMagick or GraphicsMagick.
#Creates animated plot in PDF for objects and variables
plot(res_iPCA, animation = TRUE, frames = 10, movie_format = 'pdf')


#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 saveLatex function of the animation package
#which requires ImageMagick or GraphicsMagick and
#Adobe Acrobat Reader to be installed in your system
#See help(im.convert) for details on the configuration of ImageMagick or GraphicsMagick.
#Creates animated plot in PDF movies for objects and variables
plot(res_iPCA, animation = TRUE, frames = 10, movie_format = 'pdf')
```

---

`plot.i_mca` *Plotting 2D maps in Multiple Correspondence Analysis*

---

## Description

Graphical display of Multiple Correspondence Analysis results in two dimensions

## Usage

```
## 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, movie_format = "gif", binary = FALSE,...)
```

### 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 (observations) are displayed in principal coordinates and the second entry if the variable categories are displayed in principal coordinates (default = c(TRUE,TRUE) and shows two separate plots and a joint plot if animation = FALSE and two separate plots if animation = TRUE) |
| contrib | Vector of two character strings specifying if attribute contributions should be represented by different label size. Available options are<br>"none" (contributions are not indicated in the plot)<br>"cor" (relative contributions are indicated by label size)<br>"ctr" (absolute contributions are indicated by label size)<br>The higher the contribution of a point, the larger its label size. Default is "none" |
| 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 or PDF files are created and saved to the hard drive or a static plot is created (default = TRUE) |
| frames | Number of animation frames shown per iteration (default = 10); applicable only when animation = TRUE |
| zoom | Logical indicating whether axis limits change during the animation creating a zooming effect; applicable only when animation = TRUE |
| binary | Logical indicating whether the categories associated with attribute presence are displayed on the plot; applicable only when the data are 0/1 |
| movie_format | Specifies if the animated plot is saved in the working directory either in default = "gif" or "pdf" format |
| ... | 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 pricipal coordinates.

### References

Greenacre, M.J. (1993) *Correspondence Analysis in Practice*. London: Academic Press.
Greenacre, M.J. (1993) Biplots in Correspondence Analysis, *Journal of Applied Statistics*, 20, 251-269.
ImageMagick: http://www.imagemagick.org; GraphicsMagick: http://www.graphicsmagick.org

## See Also

plot.i_pca

## Examples

```
data("women", package = "idm")
res_iMCAl = i_mca(data1 = women[1:50, 1:4], data2 = women[51:300, 1:4],
method = "live", nchunk = 4)
#static plot, final solution
plot(res_iMCAl, contrib = "ctr", animation = FALSE)

#\donttest is used here because the code calls the saveLatex function of the animation package
#which requires ImageMagick or GraphicsMagick and
#Adobe Acrobat Reader to be installed in your system
#See help(im.convert) for details on the configuration of ImageMagick or GraphicsMagick.
#Creates animated plots in PDF for objects and variables
plot(res_iMCAl, contrib = "ctr", animation = TRUE, frames = 10, movie_format = 'pdf')
```

---

# 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, movie_format = "gif", ...)
```

## 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 scatterplot of observations is displayed and the second entry if the correlation circle of the variable loadings is displayed (`default = c(TRUE,TRUE)` and shows both plots) |
| 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 or PDF files are created and saved to the hard drive or a static plot is created (default = TRUE) |
| frames | Number of animation frames shown per iteration (default = 10); applicable only when animation = TRUE |
| zoom | Logical indicating whether axes limits change during the animation creating a zooming effect; applicable only when animation = TRUE |
| movie_format | Specifies if the animated plot is saved in the working directory either in default = "gif" or "pdf" format |
| ... | 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.

## References

ImageMagick: http://www.imagemagick.org; GraphicsMagick: http://www.graphicsmagick.org

## See Also

plot.i_mca

## Examples

```
data("iris", package = "datasets")
#standardize variables
X = scale(iris[,-5])
res_iPCA = i_pca(data1 = X[1:50,-5], data2 = X[51:150,-5], nchunk = c(50,50))
#static plot, final solution
plot(res_iPCA, animation = FALSE)

##\donttest is used here because the code calls the saveLatex function of the animation package
#which requires ImageMagick or GraphicsMagick and
#Adobe Acrobat Reader to be installed in your system
#See help(im.convert) for details on the configuration of ImageMagick or GraphicsMagick.
#Creates animated plots in PDF for objects and variables
plot(res_iPCA, animation = TRUE, frames = 10, movie_format = 'pdf')
```

---

tweet *twitter data set*

---

### 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

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

### Format

A data frame with the following variables:

- `Brand` The hotel brand mentioned in the tweet: 1=`Hilton`, 2=`Intercontinental`, 3=`Marriott`, 4=`Bestwestern`, 5=`Starwood`, 6=`Hyatt`, 7=`Choice`
- `Sentiment` Sentiment for each tweet: 1=`negative (-)`, 2=`mixed (+/-)`, 3=`positive (+)`, 4=`very positive (++)`
- `UserVis` User popularity/visibility in Twitter: 1=`low`, 2=`medium`, 3=`high`

### References

Iodice D' Enza, A., & Markos, A. (2015). Low-dimensional tracking of association structures in categorical data, *Statistics and Computing*, *25*(5), 1009-1022.

Iodice D'Enza, A., Markos, A., & Buttarazzi, D. (2018). The idm Package: Incremental Decomposition Methods in R. *Journal of Statistical Software, Code Snippets*, *86*(4), 1–24. DOI: 10.18637/jss.v086.c04.

### Examples

```
data(tweet)
```

---

## `update.i_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, Lim, Lin, & Yang (2008)

### Usage

```
## S3 method for class 'i_mca'
update(object, incdata, current_rank, ff = 0, ...)
```

### Arguments

| | |
|---|---|
| `object` | object of class 'i_mca' |
| `incdata` | Matrix of incoming data |
| `current_rank` | Rank of approximation or number of components to compute; if empty, the full rank is used |
| `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 |
| `...` | Further arguments passed to `update` |

### Value

| | |
|---|---|
| `rowpcoord` | Row principal coordinates |
| `colpcoord` | Column principal coordinates |
| `rowcoord` | Row standard coordinates |
| `colcoord` | Column standard coordinates |
| `sv` | Singular values |
| `inertia.e` | Percentages of explained inertia |
| `levelnames` | Attribute names |
| `rowctr` | Row contributions |
| `colctr` | Column contributions |
| `rowcor` | Row squared correlations |
| `colcor` | Column squared correlations |
| `rowmass` | Row masses |
| `colmass` | Column masses |
| `indmat` | Indicator matrix |
| `m` | Number of cases processed up to this point |
| `ff` | A copy of `ff` in the return object |

### References

Iodice D'Enza, A., Markos, A., & Buttarazzi, D. (2018). The idm Package: Incremental Decomposition Methods in R. *Journal of Statistical Software, Code Snippets*, *86*(4), 1–24. DOI: 10.18637/jss.v086.c04.

Ross, D. A., Lim, J., Lin, R. S., & Yang, M. H. (2008). Incremental learning for robust visual tracking. *International Journal of Computer Vision*, *77*(1-3), 125–141.

### See Also

`add_es`, `i_mca`, `plot.i_mca`

### Examples

```
data(women, package = "idm")
dat = women[,c(1:4)]
res_MCA = i_mca(dat[1:300,])
aa = seq(from = 301, to = nrow(women), by = 200)
aa[length(aa)] = nrow(dat)+1
for (k in c(1:(length(aa)-1)))
{
  res_MCA = update(res_MCA,dat[c((aa[k]):(aa[k+1]-1)),])
}
plot(res_MCA, what = c(FALSE, TRUE), animation = FALSE)
```

---

## update.i_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

```
## S3 method for class 'i_pca'
update(object, incdata, current_rank, ...)
```

### Arguments

| | |
|---|---|
| `object` | object of class 'i_pca' |
| `incdata` | matrix of incoming data |
| `current_rank` | Rank of approximation or number of components to compute; if empty, the full rank is used |
| `...` | Further arguments passed to `update` |

### Value

| | |
|---|---|
| `rowpcoord` | Row scores on the principal components |
| `colpcoord` | Variable loadings |
| `eg` | A list describing the eigenspace of a data matrix, with components<br>`u` Left eigenvectors<br>`v` Right eigenvectors<br>`m` Number of cases<br>`d` Eigenvalues<br>`orgn` Data mean |
| `inertia.e` | Percentages of explained variance |

| | |
|---|---|
| sv | Singular values |
| levelnames | Variable names |
| rowcor | Row squared correlations |
| rowctr | Row contributions |
| colcor | Column squared correlations |
| colctr | Column contributions |

## References

Hall, P., Marshall, D., & Martin, R. (2002). Adding and subtracting eigenspaces with eigenvalue decomposition and singular value decomposition. *Image and Vision Computing*, *20*(13), 1009-1016.

Iodice D' Enza, A., & Markos, A. (2015). Low-dimensional tracking of association structures in categorical data, *Statistics and Computing*, *25*(5), 1009–1022.

Iodice D'Enza, A., Markos, A., & Buttarazzi, D. (2018). The idm Package: Incremental Decomposition Methods in R. *Journal of Statistical Software, Code Snippets*, *86*(4), 1–24. DOI: 10.18637/jss.v086.c04.

## See Also

update.i_mca, i_pca, i_mca, add_es

## Examples

```
data(segmentationData, package = "caret")
HCS = data.frame(scale(segmentationData[,-c(1:3)]))
names(HCS) = abbreviate(names(HCS), minlength = 5)
res_PCA = i_pca(HCS[1:200, ])
aa = seq(from = 201, to = nrow(HCS), by = 200)
aa[length(aa)] = nrow(HCS)+1
for (k in c(1:(length(aa)-1))){
     res_PCA = update(res_PCA, HCS[c((aa[k]):(aa[k+1]-1)),])
    }
#Static plot
plot(res_PCA, 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, martial 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

http://www.econ.upf.edu/~michael/women_Spain2002_original.xls

## References

Greenacre, M. J. (2010). *Biplots in practice*. Fundacion BBVA.

### Examples

```
data(women)
```

# Index