Package ‘rCUR’

February 20, 2015

Type Package
Title CUR decomposition package
Version 1.3
Date 2012-03-06
Author Andras Bodor, Norbert Solymosi
Maintainer Andras Bodor <bodri.mh3@gmail.com>
Depends R (>= 2.10.0), MASS, methods, Matrix, lattice
Description Functions and objects for CUR matrix decomposition.
License GPL (>= 2)
LazyLoad yes
Repository CRAN
Date/Publication 2012-07-02 13:20:21
NeedsCompilation no

R topics documented:

rCUR-package ......................................................... 2
CUR ................................................................. 3
CURobj-class ....................................................... 5
getC ................................................................. 6
getError ............................................................ 7
getR ................................................................. 8
getU ................................................................. 8
leverage ............................................................. 9
plotLeverage ...................................................... 10
STTa .............................................................. 11
STTm .............................................................. 11
topLeverage ...................................................... 12

Index 14
Description

In modern data mining tasks the user has often matrices with huge number of rows and/or columns as the base of analysis. One way of the analysis of this type of huge datasets is to reduce their dimensions. Principal component analysis (PCA) is a widely used tool for such data analysis. PCA produced singular vectors are mathematical abstractions and hardly interpretable on the field from which the data are drawn. Mahoney & Drineas (2009) proposed a method the CUR matrix decomposition what decreases the dimensions as well, but the resulting matrices are interpretable on the application area. This package contains functions and objects to help doing CUR matrix decomposition.

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>rCUR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>1.0</td>
</tr>
<tr>
<td>Date:</td>
<td>2011-03-31</td>
</tr>
<tr>
<td>License:</td>
<td>GPL (&gt;=2)</td>
</tr>
<tr>
<td>LazyLoad:</td>
<td>yes</td>
</tr>
</tbody>
</table>

Author(s)

Andras Bodor, Norbert Solymosi
Maintainer: Andras Bodor <bodri.mh3@gmail.com>

References


http://www.cs.rpi.edu/~boutsc/files/AlgorithmCUR.m

Examples

library(rCUR)
library(lattice)
data(STTm)
data(STTa)
n = 27
res = CUR(STTm, 31, n, 4)

plotLeverage(res, C=FALSE, top.n=n, xlab='Gene ID', las=1, top.col='black',
             top.pch=16, ul.col='black', ul.lty=2, col='grey')

PCA01 = prcomp(STTm, scale=TRUE)
top.idx = topLeverage(res, C=FALSE, top.n=n, sort=FALSE)
top.STTm = STTm[top.idx,]
PCA02 = prcomp(top.STTm, scale=TRUE)
t.top.STTm = t(top.STTm)
x = 1:dim(t.top.STTm)[1]
y = 1:dim(t.top.STTm)[2]

ann.tab = STTa[top.idx,]
genes = as.character(ann.tab$Gene)
genes[genes=='' | genes=='''' ''Hs'']

op = par(mar=c(3.5, 0, 0, 5) + 0.1)
image(x, y, t.top.STTm, col=rev(gray(0:256/300)), axes=FALSE, xlab='',
ylab='')
axis(1, at=c(10, 22, 31), labels=c('GIST', 'LEIO', 'SARC'), tick=FALSE,
las=2, cex=0.6, line=-0.8)
axis(4, at=1:length(genes), labels=genes, tick=FALSE, las=1, cex=0.6,
     line=-0.8)
par(op)

PCAs = rbind(PCA01$rotation[,c(1,2)], PCA02$rotation[,c(1,2)])
group = rownames(PCAs)
pchs = ifelse(group=='GIST', 16, 0)
pchs = ifelse(group=='LEIO', 17, pchs)
pchs = ifelse(group=='SARC', 18, pchs)
PCAs = data.frame(PC1=PCAs[,1], PC2=PCAs[,2],
                  lab=rep('all',31), rep('selected',31), group, pchs)

xyplot(PC2 ~ PC1 | lab, data = PCAs, pch=pchs, cex=1.0, col='black',
       fill='black', key = list(space = "bottom", text = list(levels(PCAs$group)),
       points=TRUE, pch=c(16, 17, 18), fill='black', columns=3, just=1))

CUR  Function to perform CUR matrix decomposition

Description

By the function the CUR matrix decomposition can be done obtaining a CURobj-class object
CUR

Usage

CUR(A, c=dim(A)[2], r=dim(A)[1], k=NULL, sv=NULL, method="random", alpha=1, weighted=FALSE, beta=4, matrix.return=TRUE, error.return=FALSE)

Arguments

A  a matrix for decomposition with m rows and n columns

C  column number to be selected from matrix A. Default: all columns, in this case column selection is skipped.

R  row number to be selected from matrix A. Default: all rows, in this case row selection is skipped.

K  rank parameter with perhaps k « min(m,n). Default: if not supplied, singular values accounting for 80% of the sum of the singular values is selected.

Sv the singular value decomposition of A. It is the most expensive part of the computation, so it can be supplied, if already available. Default: svd is computed on the fly.

Method the method, used, to select the rows. Possible values are

random the original method in [Mahoney and Drineas], rows and columns are selected randomly, with the probability of selection proportional to the leverage score

exact.num.random like random, but it is guaranteed, that exactly r rows and c columns are selected

top.scores the rows and columns with the highest leverage scores are returned deterministically

ortho.top.scores those rows and columns are selected, where the linear combination of the leverage score and orthogonality to the subspace of previously selected items is maximal. See parameter alpha.

highest.ranks rows and columns with the highest rank of leverage score for some rank parameter are selected. Every possible value is tried up to the value of k. k must be larger than 1.

Alpha if method="ortho.top.scores", the coefficient of orthogonality in the linear combination. alpha=0 is equivalent to method="top.scores". The coefficient of the leverage score is always 1. Default: 1. Should be positive.

Weighted if true, leverage scores are computed with weighting by the singular values. In this case k should be set to its default value. Not used, if method=highest.ranks. Best used with method=top.scores. See parameter beta. Default: FALSE.

Beta if weighted=TRUE, leverage scores are computed with weighting of the singular values raised to the power of beta. Default: 4.

Matrix.return if TRUE, the matrices C, U, R are returned. If matrix.return is FALSE, U is not computed, which can be expensive, if r and c are large. Default: TRUE.

Error.return if true, the Frobenius norm of the difference between the original matrix and the CUR approximation is returned. Effective only if matrix.return is TRUE. Default: FALSE.
The function produces an object of \texttt{CURobj-class}.

\section*{References}


The development was initially based on the Matlab code of Christos Boutsidis:
http://www.cs.rpi.edu/~boutsc/files/AlgorithmCUR.m

\section*{Examples}
\begin{verbatim}
data(STTm)
CUR.results <- CUR(STTm, 31, 12, 3)
\end{verbatim}

---

\texttt{CURobj-class} \hspace{1cm} \textit{Class "CURobj"}

\section*{Description}
Class for the results of CUR decomposition performed by the function \texttt{CUR}

\section*{Objects from the Class}
Objects can be created by calls to the function \texttt{CUR}

\section*{Slots}
\begin{itemize}
\item \texttt{C}: matrix \texttt{C}
\item \texttt{U}: matrix \texttt{U}
\item \texttt{R}: matrix \texttt{R}
\item \texttt{C.leverage.score}: leverage scores of the columns
\item \texttt{R.leverage.score}: leverage scores of the rows
\item \texttt{C.index}: index of columns selected for \texttt{C}
\item \texttt{R.index}: index of rows selected for \texttt{R}
\item \texttt{Error}: the error of the approximation
\end{itemize}
Methods

- **getC** retrieves the matrix C from object of CURobj-class
- **getU** retrieves the matrix U from object of CURobj-class
- **getR** retrieves the matrix R from object of CURobj-class
- **getError** retrieves the error from object of CURobj-class
- **leverage** retrieves the vector of leverage scores from object of CURobj-class
- **topLeverage** retrieves the top elements of leverage scores from object of CURobj-class
- **plotLeverage** plot the leverage scores of CURobj-class object

References


Examples

```
showClass("CURobj")
```

---

**getC**

*Retrieve matrix C from CURobj-class*

Description

Retrieve matrix C from CURobj-class

Usage

```
getC(object)
```

Arguments

- **object** object of class CURobj-class

Value

a matrix with component C of CUR

References


**getError**

**Examples**

```r
data(STTm)
res <- CUR(STTm, 31, 12, 3)
getC(res)
```

---

**getError**

*Retrieve the error of the approximation from CURobj-class*

**Description**

Retrieve the error of the approximation from CURobj-class

**Usage**

```r
getError(object)
```

**Arguments**

- **object** object of class CURobj-class

**Value**

The Frobenius norm of the difference between the original matrix and the CUR approximation, or NULL, if CUR was called with error.return=FALSE (default).

**References**


**Examples**

```r
data(STTm)
res <- CUR(STTm, 31, 12, 3, error.return=TRUE)
getError(res)
```
getR

Retrieve matrix R from CURobj-class

Description

Retrieve matrix R from CURobj-class

Usage

getR(object)

Arguments

object object of class CURobj-class

Value

a matrix with component R of CUR

References


Examples

data(STTm)
res <- CUR(STTm, 31, 12, 3)
getR(res)

getU

Retrieve matrix U from CURobj-class

Description

Retrieve matrix U from CURobj-class

Usage

getU(object)

Arguments

object object of class CURobj-class
leverage

Value

a matrix with component U of CUR

References


Examples

data(STTm)
res <- CUR(STTm, 31, 12, 3)
getU(res)

leverage

Retrieve leverage scores from CURobj-class

Description

Retrieve leverage scores from CURobj-class

Usage

leverage(object, C = TRUE)

Arguments

object object of class CURobj-class
C logical, if it is TRUE, then the obtained leverage scores belongs to columns, if FALSE, then to rows

Value

the function returns a numerical vector of leverage scores

References


Examples

data(STTm)
res <- CUR(STTm, 31, 12, 3)
leverage(res, C=TRUE)
Function to plot leverage scores

Description

Function to plot the individual values and uniform level of leverage scores. The most influential features may be highlighted.

Usage

plotLeverage(x, C=TRUE, mplr=1000, top.n=100, top.col='red', top.pch=16, ul.col='red', ul.lty=2, ...)

Arguments

x object of class CURobj-class
C logical, if it is TRUE, then calculated the rank for the leverage scores belongs to columns, if FALSE, then to rows
mplr multiplier of leverage scores
top.n number of highlighting items from top of leverage score list
top.col color of highlighting points
top.pch pch of highlighting points, see par()
ul.col color of the line of leverage score uniform level
ul.lty line type of leverage score uniform level
... further graphical parameters of function plot()

References


Examples

data(STM)
res <- CUR(STM, 31, 12, 3)
plotLeverage(res, C=FALSE, top.n=10, las=1)
Description

Annotation data for STTm soft tissue tumour gene expression dataset

Format

A data frame with 5520 observations on the following 4 variables.

- **id**: identification number of gene
- **gene**: gene symbol
- **Hs**: UniGene ID
- **Descr**: function description of transcriptum of gene

Source

The original, full dataset is downloadable from the Gene Expression Omnibus database (GSE3443) and from Stanford Microarray Database (http://smd.stanford.edu).

References


Examples

```r
data(STTm)
```

Description

STTm is a matrix containing a subset of the soft tissue tumour gene expression study dataset published by Nielsen et al. (2002). The matrix constructed from 31 columns and 5520 rows, samples and genes respectively. Samples came from three different tumour types: gastrointestinal stromal tumor (GIST), leiomyosarcoma (LEIO) and synovial sarcoma (SARC).
topLeverage

Source

The original, full dataset is downloadable from the Gene Expression Omnibus database (GSE3443) and from Stanford Microarray Database (http://smd.stanford.edu).

References


Examples

data(STTm)

topLeverage  Function to obtain indexes of the most influential features due to leverage score

Description

The function sort the features (rows or columns) based on their influence (leverage score) and list out the indexes of the top valued items

Usage

topLeverage(object, C=TRUE, top.n=100, sort=TRUE)

Arguments

  object object of class CURobj-class
  C logical, if it is TRUE, then calculated the rank for the leverage scores belongs to columns, if FALSE, then to rows
  top.n number of indexes to list
  sort logical, if TRUE the indexes are sorted increasing

Value

The function returns a numerical vector of indexes of the most influential features
topLeverage

References


Examples

data(STTm)
res <- CUR(STTm, 3, 31, 12)
topLeverage(res, top.n=10)
Index

*Topic **classes**
  CURobj-class, 5

*Topic **datasets**
  STTa, 11
  STTm, 11

*Topic **prcomp**
  CUR, 3
  rCUR-package, 2

*Topic **svd**
  CUR, 3
  rCUR-package, 2

CUR, 3, 5
CURobj-class, 3, 5, 6–10, 12

getC, 6, 6
getC, CURobj-method (CURobj-class), 5
getError, 6, 7
getError, CURobj-method (CURobj-class), 5
getR, 6, 8
getR, CURobj-method (CURobj-class), 5
getU, 6, 8
getU, CURobj-method (CURobj-class), 5

leverage, 6, 9
leverage, CURobj-method (CURobj-class), 5

plotLeverage, 6, 10
plotLeverage, CURobj-method
  (CURobj-class), 5

rCUR (rCUR-package), 2
rCUR-package, 2

STTa, 11
STTm, 11

topLeverage, 6, 12
topLeverage, CURobj-method
  (CURobj-class), 5