Package 'ExpressionView'

November 18, 2019

Version 1.39.0

Date Sep 23, 2012

Title Visualize biclusters identified in gene expression data

Author Andreas Luscher <andreas.luescher@a3.epfl.ch>

Maintainer Gabor Csardi <csardi.gabor@gmail.com>

Description ExpressionView visualizes possibly overlapping biclusters
in a gene expression matrix. It can use the result of the ISA
method (eisa package) or the algorithms in the biclust
package or others. The viewer itself was developed using Adobe
Flex and runs in a flash-enabled web browser.

Depends caTools, bitops, methods, isa2, eisa, GO.db, KEGG.db,
AnnotationDbi

Imports methods, isa2, eisa, GO.db, KEGG.db, AnnotationDbi

Suggests ALL, hgu95av2.db, biclust, affy

biocViews Classification, Visualization, Microarray, GeneExpression,
GO, KEGG

Collate AllGenerics.R export.R launch.R order.R zzz.R

License GPL (>= 2)

git_url https://git.bioconductor.org/packages/ExpressionView

git_branch master

git_last_commit b9df7fc

git_last_commit_date 2019-10-29

Date/Publication 2019-11-17

R topics documented:

ExpressionView-package

The ExpressionView package

Description

A package designed to interactively explore biclusters identified in gene expression data.

Introduction

Clustering genes according to their expression profiles is an important task in analyzing microarray data. ExpressionView allows the user to explore the biclusters together with the underlying data in an interactive environment. The applet requires a browser with Adobe Flash player 10 installed.

The ExpressionView package can treat gene expression data in the form of a Bioconductor ExpressionSet. It recognizes biclusters identified by the Iterative Signature Algorithm (ISA) implemented in the ISA package as well as the methods available in the biclust package.

Worflow

The usual workflow consist of three steps:

Order To arrange the possibly overlapping biclusters in a visually appealing layout, the gene expression data has to be reordered in such a way that individual biclusters from contiguous rectangles. The optimal order is found by the OrderEV function.

Export In a second step, the biclusters and all the relevant biological information are combined and exported to an ExpressionView XML file. This is done by the ExportEV function.

Visualize The Flash applet is started by the LaunchEV command. Video tutorials describing the various features of the applet can be found on the ExpressionView website (http://www.unil.ch/cbg/ExpressionView).

A tutorial can be found in the accompanying vignette of the package.

Biclustering with non-gene expression data

Both, the ISA and the biclustering methods implemented in the biclust package can treat any two-dimensional data, i.e., not necessarily originating from gene expression profiling. While the ExpressionView package is optimized for gene expression matrices, it is also possible to explore data stemming from other sources. For more information, see the description of the ExportEV function.

Author(s)

Andreas Lüscher <andreas.luescher@a3.epfl.ch>

See Also

OrderEV, ExportEV and LaunchEV

ExportEV   *Export an ExpressionView file*

Description

Exports the biclusters identified in gene expression data with all the relevant biological data to an XML file that can be read by the ExpressionView Flash applet.

Usage

```
## S4 method for signature 'ISAModules'
ExportEV(biclusters, eset,
        order=OrderEV(biclusters), filename=file.choose(),
        norm=c("sample", "feature", "raw", "x", "y"), cutoff=0.95,
        description=NULL, GO, KEGG, ...)
## S4 method for signature 'Biclust'
ExportEV(biclusters, eset, order, filename, norm,
        cutoff, description, ...)
## S4 method for signature 'list'
ExportEV(biclusters, eset, order=OrderEV(biclusters),
        filename=file.choose(),
        norm=c("sample", "feature", "raw", "x", "y"),
        cutoff=0.95, description=NULL, ...)
```

Arguments

biclusters	An [ISAModules](#) object, a [Biclust](#) object, or a named list, the last one possibly coming from the isa2 package.
eset	A [ExpressionSet](#) object containing the gene expression data. Please see below how to use this function on other kind of data.
order	A named list (result of the [OrderEV](#) function) containing the optimal order. If not specified, an ordering with the default parameters is performed.
filename	The filename of the output file. If not specified, the file is selected via the user interface.
norm	The normalization of the gene expression data. The [isa.normalize](#) function can normalize (zero mean and unit variance) the data with respect to the genes or the samples. Possible values: 'feature', 'sample' and 'raw'. 'x' is the same as 'feature' and 'y' is the same as 'sample'. The default value is 'sample'.
cutoff	The cutoff for the coloring is a value between 0 and 1. It represents the fraction of data points taken into account for the density plots. The default value is 0.95, i.e., the extrema of the coloring are chosen in such a way that 95% of the data points can be represented.
description	A named list containing an alternative description of the data. By default, the metadata is extracted from eset. Please see below how to assemble the data description if you are dealing with data other than gene expression.
GO	A list of three GOListHyperGResult objects, containing the enrichment calculation results for the three Gene Ontology ontologies, for all modules, as returned by the ISAGO function in the eisa package. If not specified, then it is calculated automatically.

KEGG	A `GOListHyperGResult` object, that contains the of the enrichment calculation results for all modules, against the KEGG pathway database, as returned by the `ISAKEGG` function in the `eisa` package. If not specified, then it is calculated automatically.
...	Additional arguments, nothing currently.

Details

If the data is available in the form of a `ExpressionSet`, the `ExportEV` function automatically uses the metadata associated with the gene expression data. If the underlying data does not contain any annotations, you can provide them manually, by defining various items in the description list, see the second example below.

Author(s)

Andreas Lüscher <andreas.luescher@a3.epfl.ch>

See Also

`OrderEV`, `LaunchEV`, `ISA`, `biclust`

Examples

```
## Gene expression data
## We use the acute T-cell lymphocytic leukemia (ALL) data together with
## the Iterative Signature Algorithm (ISA).

## Load the package and the ALL data
library(ExpressionView)
library(eisa)
library(ALL)
library(hgu95av2.db)
data(ALL)

## Initialize random number generator to get reproducible results
set.seed(5)

## Find biclusters (=modules)
## To avoid some minutes of waiting, we just load the data
## set included in the 'eisa' package instead of
## really performing the calculation.
#modules <- ISA(ALL, thr.gene=2.7, thr.cond=1.4)
data(ALLModulesSmall)
modules <- ALLModulesSmall

## Realign the gene exptression matrix to optimize arrangements of
## biclusters
optimalorder <- OrderEV(modules)

## Export the data to an ExpressionView file
## Don't forget to change the filename
## Not run: ExportEV(modules, ALL, optimalorder, filename="file.evf")


## In-silico data
## We use insilico data together with the ISA and manually annotate the
```

```
## data set. Simply explore the data file with the Flash applet to
## figure out where the various annotations are placed.

## Load the package
library(ExpressionView)

## Generate noisy in-silico data with dimensions m x n
m <- 50
n <- 500
data <- isa.in.silico(num.rows=m, num.cols=n, noise=0.1,
                      overlap.row=0)[[1]]

## Find biclusters (=modules)
modules <- isa(data)

## Annotate the rows and columns of data set
rownames(data) <- paste("row", seq_len(nrow(data)))
colnames(data) <- paste("column", seq_len(ncol(data)))

## Add metadata associated with the rows of the data set
rowdata <- outer(1:nrow(data), 1:sample(1:20, 1), function(x, y) {
  paste("row description (", x, ", ", y, ")", sep="")
})
rownames(rowdata) <- rownames(data)
colnames(rowdata) <- paste("row tag", seq_len(ncol(rowdata)))

## Add metadata associated with the columns of the data set
coldata <- outer(1:ncol(data), 1:sample(1:20, 1), function(x, y) {
  paste("column description (", x, ", ", y, ")", sep="")
})
rownames(coldata) <- colnames(data)
colnames(coldata) <- paste("column tag", seq_len(ncol(coldata)))

## Merge the different annotations in a single list and
## add a few global things
description <- list(
experiment=list(
title="Title",
xaxislabel="x-Axis Label",
yaxislabel="y-Axis Label",
name="Author",
lab="Address",
abstract="Abstract",
url="URL",
annotation="Annotation",
organism="Organism"),
coldata=coldata,
rowdata=rowdata
)

## Realign the gene exptression matrix to optimize arrangements of
## biclusters
optimalorder <- OrderEV(modules)

## Export the data to an ExpressionView file
## Don't forget to change the filename
ExportEV(modules, data, optimalorder, filename="file.evf",
```

```
        description=description)
```

LaunchEV   *Launch the ExpressionView Flash applet*

Description

Launches the ExpressionView Flash applet that comes with the R package.

Usage

```
    LaunchEV()
```

Details

LaunchEV opens the ExpressionView Flash applet for visualizing biclusters identified in gene expression data. Running the applet requires a Browser with Adobe Flash 10 installed. The program is written in ActionScript and Adobe Flex. The source code is freely available. You can get it by right-clicking into the applet and selecting "View Source".

Author(s)

Andreas Lüscher <andreas.luescher@a3.epfl.ch>

See Also

OrderEV, ExportEV, ISA, biclust

Examples

```
    ## Launch the ExpressionView applet
    ## Not run: LaunchEV()
```

OrderEV   *Find the optimal arrangement of biclusters for visualization in ExpressionView*

Description

Finds the optimal arrangement of possibly overlapping biclusters that maximizes the areas of the largest contiguous parts of the biclusters. The reordering is necessary to obtain a visually appealing layout of the biclusters.

Usage

```
    ## S4 method for signature 'ISAModules'
    OrderEV(biclusters, initialorder, maxtime, debuglevel)
    ## S4 method for signature 'Biclust'
    OrderEV(biclusters, initialorder, maxtime, debuglevel)
    ## S4 method for signature 'list'
    OrderEV(biclusters, initialorder, maxtime, debuglevel)
```

Arguments

biclusters	An ISAModules object, a Biclust object or a named list. The last one is probably coming from the isa2 package.
initialorder	A list containing the initial order. Usually the output of a previous ordering.
maxtime	The maximal computation time in seconds. The default value is one minute (maxtime=60).
debuglevel	The level of information provided during the ordering. By default, the debug output is turned off (debuglevel=0).

Details

OrderEV performs a brute-force ordering of the biclusters, treating the rows and the columns of the matrix independently. The ordering algorithm is described in more detail in the accompanying vignette of the package.

Value

A named list is returned with the following elements:

rows / genes	A list containing the maps between the rows of the initial and the optimally ordered gene expression matrix. The first element represents the map of the complete data set, while the subsequent entries contain the row maps of the data sets projected onto the individual clusters. This entry is called 'rows' if the function is called with a simple list as the first argument, and 'genes' otherwise.
cols / samples	A list containing the maps between the columns of the initial and the optimally ordered gene expression matrix. The first element represents the map of the complete data set, while the subsequent entries contain the column maps of the data sets projected onto the individual clusters. This entry is called 'cols' if the function is called with a simple list as the first argument, and 'samples' otherwise.
status	A list containing the status of the ordering. The list has two entries, named genes and samples (or rows and cols if the function was called with a simple list as the first argument). Each entry is a numeric vector of ones and zeros. A 1 indicates that the map is fully optimized, whereas a 0 signals that the ordering could not be completely within the given time frame.

Author(s)

Andreas Lüscher <andreas.luescher@a3.epfl.ch>

See Also

ExportEV, LaunchEV, ISA, biclust

Examples

```
## We generate some noisy in-silico data with biclusters,
## scramble the initially ordered arrangement
## identify the bicluster with the Iterative Signature Algorithm (ISA)
## and order the results with the OrderEV function
library(isa2)
data.in.silico <- isa.in.silico(noise=0.1)[[1]]
```

```
data.in.silico <- data.in.silico[sample(c(1:dim(data.in.silico)[1])),
                                  sample(c(1:dim(data.in.silico)[2]))]
isa.results <- isa(data.in.silico)
optimalorder <- OrderEV(isa.results)
str(optimalorder)

## Create a plot for the scrambled and the optimal orderings
## Not run:
layout(rbind(1:2))
image(data.in.silico)
image(data.in.silico[optimalorder$rows[[1]],
                     optimalorder$cols[[1]]])

## End(Not run)
```

Index