BIOINFORMATICS

Vol. 00 no. 00 2009
Pages 1–2

Modular analysis of gene expression data with GNU R

Gábor Csárdi ,[1,2], Zoltán Kutalik [1,2] and Sven Bergmann [1,2]

[1]Department of Medical Genetics, and [2]Swiss Institute of Bioinformatics, University of Lausanne, Rue de Bugnon 27, CH-1005 Lausanne, Switzerland.

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Associate Editor: XXXXXXX

ABSTRACT

TODO

Summary: TODO

Availability: http://www.unil.ch/cbg/homepage/software.html

Contact: Sven.Bergmann@unil.ch

1 INTRODUCTION

Biclustering is the idea of clustering the rows and the columns of a data matrix together, usually not independently. In other words, the goal of a biclustering algorithm is to find blocks in the reordered data matrix that have, such that the blocks have correlated rows and columns. It is important to note that these blocks (or biclusters, modules) have context, i.e. the rows only need to be correlated across the columns of the block, but not across other columns of the data matrix; and vice-versa.

When it comes to gene expression data, a bicluster means a subset of genes (G) (or probes, probe sets) and a subset of samples (S), such that genes G are exactly coexpressed across S and samples S have correlated expression profile exactly across G.

The Iterative Signature Algorithm (ISA), Ihmels *et al.* (2002); Bergmann *et al.* (2003); Ihmels *et al.* (2004), is a powerful biclustering algorithm. It is able to identify biclusters that overlap and it is resilient to noise.

In this short note, we introduce the '`isa2`' and '`eisa`' GNU R, R Development Core Team (2009), packages; these implement the ISA biclustering method, provide a convenient interface to run it using the BioConductor, Gentleman *et al.* (2004), software package, and also contain visualization tools.

2 METHODS

Let us first explain the briefly the steps of a typical modular analysis study for gene expression data.

Batch correction Often, the ISA is applied to samples from different experiments. In this case it is crucial to address the question of experimental variation. Several methods exist to solve this problem, see e.g. Johnson *et al.* (2007) for an algorithm that has a GNU R implementation.

Gene filtering Depending on the application, it can be advantageous to remove genes that are not expressed in any of the samples, especially because the ISA normalization (next step) tends to amplify their effect.

ISA normalization ISA is an iterative algorithm. In each step it computes weighted sums of expression levels for a number of genes or a number of samples. Since different genes typically show different levels of base expression and variance, it is important to standardize expression levels to Z-scores. The ISA uses two sets of Z-scores, one calculated for each gene across all samples, the other for each sample across all genes.

Random and smart seeding The ISA used with random seeds is an unsupervised algorithm, it uses no external knowledge to find the biclusters. ISA with smart seeds is biased towards genes or pathways of interest, or sample annotation. This latter form can be considered as a semi-supervised approach.

The ISA iteration This step is essentially performing the ISA and finding the biclusters from the starting seeds.

Merging the modules It is possible that more seeds converge to the same, or very similar biclusters. This step eliminates such duplicates.

Robustness of biclusters To access the significance of a bicluster, we designed a robustness measure, that can be used to filter out spurious modules. This is done based on performing ISA on the scrambled input data.

Module trees The ISA works with two stringency threshold parameters, the gene threshold and the sample threshold. ISA modules can be organized into a directed graph, a module tree, in which there is an edge from module A to module B, if the ISA converges to module B from module A, with the same threshold parameters that were used to find module B. A module tree provides a hierarchical modular description of a data set.

3 IMPLEMENTATION

The ISA and accompanying visualization tools are implemented in two R packages. The '`isa2`' package contains the implementation of the basic ISA itself; this package can be used to analyze any tabular data. The '`eisa`' package builds on '`isa2`'. It adds support to standard BioConductor gene expression data structures; and contains gene expression specific visualization tools, see some of these in Fig. 1.

Both the '`isa2`' and '`eisa`' packages support two workflows. The *simple workflow* involves a single R function call and runs all ISA steps (1-3 on Fig. 1.) with their default parameters.

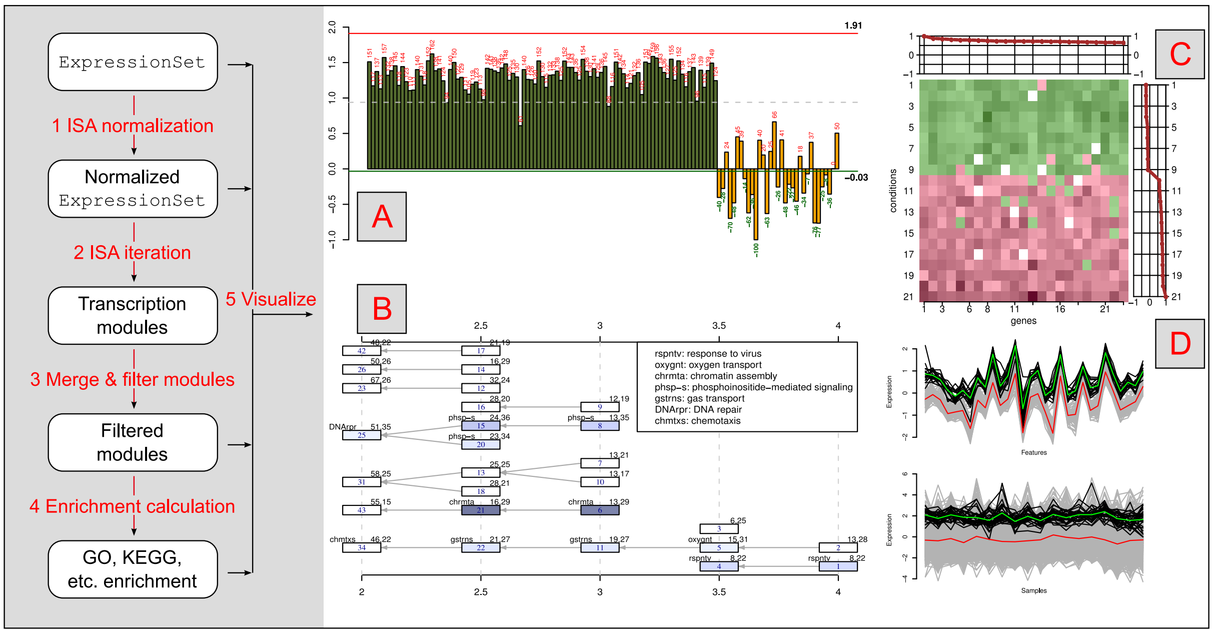

Fig. 1. Work flow of a typical modular analysis study, with the ‘eisa’ package (left); and four bicluster visualization techniques (right). The input of the ‘eisa’ pipeline is a BioConductor *ExpressionSet* object. This is normalized to Z-scores (1); the ISA iteration is performed on it using smart or random seeds (2); and the biclusters are then merged and filtered based on the robustness measure (3). The ‘eisa’ package contains tools to perform many GO, KEGG, etc. enrichment calculation quickly (4), and several bicluster visualization tools (5). Subfigures **A**, **B**, **C** and **D** were generated using the acute lymphoblastic leukemia data set, see Chiaretti *et al.* (2004) and the ‘ALL’ R package. **(A)** A condition plot for a simple bicluster. Each sample is represented as a bar, its height is its score in the bicluster. Samples above the red (none in this case) and samples below the green line are part of the biclusters. The coloring indicates cases and controls. This module can separate control and case samples. **(B)** A module tree. Each rectangle is a bicluster. See the definition of the module tree in the text. The modules are colored according to their Gene Ontology enrichment p-values, the codes of the enriched GO categories are shown in the top-left corner of the rectangles. The top-right corner shows the number of genes and conditions in the bicluster. The numbers on the horizontal axes are the gene thresholds used for finding the modules. **(C)** Heatmap for a single module, showing correlated expression and the genes and samples. The red lines are the gene and sample scores. **(D)** Profile plot, for a single module. On the top plot the expression of all samples are plotted, the samples of the module with black. The red and green lines show the mean of the two groups. The bottom plot is the same for the genes of the module.

The *detailed workflow* means running every step of the modular analysis separately, possibly with non-default parameters. This allows the users to tailor the ISA completely according their needs.

For users that prefer Matlab over R, we created a Matlab package for ISA, please see the ISA homepage for more information.

The ‘eisa’ package implements a set of visualization techniques for biclusters, some of these are specific to ISA, others are not. Please see Fig. 1 for some of these.

The ‘ExpressionView’ R package implements an interactive tool for visualizing ISA (or other) overlapping biclusters.

The ‘biclust’ package, Kaiser *et al.* (2009), implements a number of biclustering algorithms in a unified framework. The ‘eisa’ package include tools to convert between ‘biclust’ and ISA biclusters. This allows the cross-talk of the functions in the two packages.

ACKNOWLEDGEMENT

TODO

Funding: TODO

Conflict of interest: None declared.

REFERENCES

Bergmann, S., Ihmels, J., and Barkai, N. (2003). Iterative signature algorithm for the analysis of large-scale gene expression data. *Phys Rev E Nonlin Soft Matter Phys*, page 031902.

Chiaretti, S., Li, X., Gentleman, R., Vitale, A., Vignetti, M., Mandelli, F., Ritz, J., and Foa, R. (2004). Gene expression profile of adult t-cell acute lymphocytic leukemia identifies distinct subsets of patients with different response to therapy and survival. *Blood*, **103**(7).

Gentleman, R. C., Carey, V. J., Bates, D. M., *et al.* (2004). Bioconductor: Open software development for computational biology and bioinformatics. *Genome Biology*, **5**, R80.

Ihmels, J., Friedlander, G., Bergmann, S., Sarig, O., Ziv, Y., and Barkai, N. (2002). Revealing modular organization in the yeast transcriptional network. *Nat Genet*, pages 370–377.

Ihmels, J., Bergmann, S., and Barkai, N. (2004). Defining transcription modules using large-scale gene expression data. *Bioinformatics*, pages 1993–2003.

Johnson, W., Rabinovic, A., and Li, C. (2007). Adjusting batch effects in microarray expression data using empirical bayes methods. *Biostatistics*, **8**(1), 118–127.

Kaiser, S., Santamaria, R., Theron, R., Quintales, L., and Leisch., F. (2009). *biclust: BiCluster Algorithms*. R package version 0.8.1.

R Development Core Team (2009). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.