BicARE : Biclustering Analysis and Results Exploration

Pierre Gestraud\textsuperscript{1,2,3}, Isabel Brito\textsuperscript{1,2,3} and Emmanuel Barillot\textsuperscript{1,2,3}

October 24, 2023

1. Institut Curie, Paris, F-75248 France
2. INSERM, U900, Paris, F-75248 France
3. Ecole des Mines de Paris, Fontainebleau, F-77300 France

http://bioinfo.curie.fr

Contents

1 Overview  1

2 Biclustering analysis  1

2.1 Random initialisation  2

2.2 Directed initialisation  3

2.3 Bicluster extraction  3

3 Additional analyses  4

3.1 Genesets enrichment  4

3.2 Sample covariates enrichment  5

4 Building html report  5

1 Overview

This document presents an overview of the BicARE package. This package is dedicated to biclustering analysis which allows to discover sets of genes that have the same expression pattern across a set of samples (for an overview of bicluster analysis see Madeira and Oliveira (2004)).

2 Biclustering analysis

Data set used in this vignette is a subset of the sample.ExpressionSet, 26 hgu95av2 arrays normalised by dChip. Only 352 probesets are kept (probesets with a minimal value greater than 1) and expression values are set to log2 scale.

> data(sample.bicData)

[1] "sample.bicData"

> sample.bicData
ExpressionSet (storageMode: lockedEnvironment)
assayData: 352 features, 26 samples
  element names: exprs, se.exprs
protocolData: none
phenoData
  sampleNames: A B ... Z (26 total)
  varLabels: sex type score
  varMetadata: labelDescription
featureData: none
experimentData: use 'experimentData(object)'
Annotation: hgu95av2

The biclustering algorithm used in BicARE is based on the notion of residue which is a measure of coherence of the elements in a bicluster (see Yang et al. (2005) for a definition of the residue). The smaller the residue, the more coherent the bicluster.

Computing the residue of the data matrix :

```r
> residue(sample.bicData)
[1] 0.3401921
```

The core of the package is the FLOC function which launches a modified version of the FLOC (FLexible Ov erlapped biClustering) algorithm, see Yang et al. (2005). A predetermined number (parameter \( k \)) of biclusters are build such that they are as big as possible with a residue smaller than the threshold (parameter \( r \)).

### 2.1 Random intialisation

In a strictly exploratory approach it is possible to build the biclusters around random seeds (random biclusters that are iteratively improved). The size of the random seeds is controlled by parameters \( pGene \) and \( pSample \) such as each gene (sample) has a probability \( pGene \) (\( pSample \)) to belong to each bicluster. Other parameters are the number of biclusters build (\( k \)), the residue threshold (\( r \)), the minimal number of genes (\( N \)) per bicluster, the minimal number of conditions (\( M \)) per bicluster and the number of iterations (\( t \)).

```r
> set.seed(1)
> res.biclustering <- FLOC(sample.bicData, k=15, pGene=0.3, pSample=0.6, r=0.01, 10, 8, 200)
> res.biclustering
```

Call :
FLOC(Data = sample.bicData, k = 15, pGene = 0.3, pSample = 0.6,
  r = 0.01, N = 10, M = 8, t = 200)

Parameters :
  number of biclusters : 15
  residu threshold : 0.01
  gene initial probability : 0.3
  sample initial probability : 0.6
  number of iterations : 200
  date : Tue Oct 24 16:24:07 2023
Biclusters :
A data frame \( \text{mat.resvol.bic} \) gives the characteristics of the biclusters such as the residue or the size of the biclusters. In the rowvar column is displayed the mean of the variance of the genes in the bicluster. It helps to find non-trivial biclusters (e.g. constant bicluster).

### 2.2 Directed initialisation

It is also possible to build biclusters around previous knowledge. Initialisation can be performed on genes, on samples or on both.

The initialisation is performed by building boolean matrices indicating the membership of the elements to the biclusters.

```r
> init.genes <- matrix(data=0, nrow=352, ncol=5)
> init.samples <- matrix(data=0, nrow=26, ncol=5)
> init.genes[1:10,1] <- 1
> init.genes[20:30,2] <- 1
> init.genes[50:60,3] <- 1
> init.samples[1:5,3] <- 1
> init.samples[1:5,4] <- 1
> init.samples[10:15,5] <- 1
```

Here the five first biclusters are initialised. Biclusters 1 and 2 are respectively initialised around genes 1 to 10 and 20 to 30; biclusters 4 and 5 around samples 1 to 5 and 10 to 15; bicluster 3 is initialised around genes 50 to 60 and samples 1 to 5.

If the number of biclusters initialised is different from the parameter \( k \), the real number of biclusters searched is the greater of the two.

The two initialisation matrices are given as arguments blocGene and blocSample to the FLOC function.

### 2.3 Bicluster extraction

The bicluster function extracts a bicluster from a biclustering object. The bicluster is returned as a matrix with the genes on rows and the conditions on columns. The graph option determines if a graphic should be plotted.
3 Additionnal analyses

Additionnal analyses can be performed on the biclustering results. These analyses allows a functional view of the biclusters by testing the over-representation of gene sets or a characterisation of the samples (e.g. clinical covariates).

3.1 Genesets enrichment

A functional view of the biclusters can be obtained by testing the over-representation of a priori defined gene sets. This over-representation is evaluated by an hypergeometric test. The gene sets are in the GeneSetCollection format.

> gsc <- GeneSetCollection(res.biclustering$ExpressionSet[1:50], setType=GOCollection())
> res.bic2 <- testSet(res.biclustering, gsc)

The testSet function returns an updated biclustering object with a new attribute geneSet. It is a list containing the GeneSetCollection used, the p-values (from an hypergeometric test) and the adjusted p-values.
3.2 Sample covariates enrichment

Three covariates for the samples are provided in the example data set. Function testAnnot only uses categorical covariates (here sex and type). For each bicluster, the function evaluates, with a $\chi^2$ test of adequation, the enrichment of each level of the covariates.

> pData(sample.bicData)

<table>
<thead>
<tr>
<th></th>
<th>sex</th>
<th>type</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Female</td>
<td>Control</td>
<td>0.75</td>
</tr>
<tr>
<td>B</td>
<td>Male</td>
<td>Case</td>
<td>0.40</td>
</tr>
<tr>
<td>C</td>
<td>Male</td>
<td>Control</td>
<td>0.73</td>
</tr>
<tr>
<td>D</td>
<td>Male</td>
<td>Case</td>
<td>0.42</td>
</tr>
<tr>
<td>E</td>
<td>Female</td>
<td>Case</td>
<td>0.93</td>
</tr>
<tr>
<td>F</td>
<td>Male</td>
<td>Control</td>
<td>0.22</td>
</tr>
<tr>
<td>G</td>
<td>Male</td>
<td>Case</td>
<td>0.96</td>
</tr>
<tr>
<td>H</td>
<td>Male</td>
<td>Case</td>
<td>0.79</td>
</tr>
<tr>
<td>I</td>
<td>Female</td>
<td>Case</td>
<td>0.37</td>
</tr>
<tr>
<td>J</td>
<td>Male</td>
<td>Control</td>
<td>0.63</td>
</tr>
<tr>
<td>K</td>
<td>Male</td>
<td>Case</td>
<td>0.26</td>
</tr>
<tr>
<td>L</td>
<td>Female</td>
<td>Control</td>
<td>0.36</td>
</tr>
<tr>
<td>M</td>
<td>Male</td>
<td>Case</td>
<td>0.41</td>
</tr>
<tr>
<td>N</td>
<td>Male</td>
<td>Case</td>
<td>0.80</td>
</tr>
<tr>
<td>O</td>
<td>Female</td>
<td>Case</td>
<td>0.10</td>
</tr>
<tr>
<td>P</td>
<td>Female</td>
<td>Control</td>
<td>0.41</td>
</tr>
<tr>
<td>Q</td>
<td>Female</td>
<td>Case</td>
<td>0.16</td>
</tr>
<tr>
<td>R</td>
<td>Male</td>
<td>Control</td>
<td>0.72</td>
</tr>
<tr>
<td>S</td>
<td>Male</td>
<td>Case</td>
<td>0.17</td>
</tr>
<tr>
<td>T</td>
<td>Female</td>
<td>Case</td>
<td>0.74</td>
</tr>
<tr>
<td>U</td>
<td>Male</td>
<td>Control</td>
<td>0.35</td>
</tr>
<tr>
<td>V</td>
<td>Female</td>
<td>Control</td>
<td>0.77</td>
</tr>
<tr>
<td>W</td>
<td>Male</td>
<td>Control</td>
<td>0.27</td>
</tr>
<tr>
<td>X</td>
<td>Male</td>
<td>Control</td>
<td>0.98</td>
</tr>
<tr>
<td>Y</td>
<td>Female</td>
<td>Case</td>
<td>0.94</td>
</tr>
<tr>
<td>Z</td>
<td>Female</td>
<td>Case</td>
<td>0.32</td>
</tr>
</tbody>
</table>

> res.bic2 <- testAnnot(res.biclustering, annot=pData(sample.bicData), covariates=c("sex", "type"))

As testSet does, the testAnnot function returns an updated biclustering object with a new attribute covar. It is a list containing the covariates used, the p-values and adjusted p-values, the numbers of each level in each bicluster and the residuals of the tests.

4 Building html report

Dealing with biclustering results is a tedious task because of the amount of data to explore. A user-friendly report can be created with the function makeReport. It allows an easy navigation through the results by different approaches:

- by bicluster
- by gene sets in order to examine the biclusters enriched in gene sets of interest
• by sample covariates in order to examine the biclusters enriched in samples with characteristics of interest

Each bicluster is presented by the lists of the genes and samples and by a plot of the expression profiles.

```r
> dirPath <- getwd()
> dirName <- "test"
> makeReport(dirPath=dirPath, dirName=dirName, resBic=res.bic2, browse=FALSE)
```

The html report is created in the directory `dirName` in the current working directory. The `browse` option determines if the web browser is opened on the `home.html` file. Before running `makeReport` it is advised to check if the results directory can be created.

**References**
