Journal of Proteomics & Bioinformatics

ISSN: 0974-276X

Saber and Elloumi, J Proteomics Bioinform 2015, S9
DOI: 10.4172/jpb.S9-008

Research Article Open Access

Efficiently Mining Gene Expression Data via Novel Binary Biclustering Algorithms

Haifa Ben Saber[1]* and Mourad Elloumi[1,2]

[1]*Laboratory of Technologies of Information and Communication and Electrical Engineering (LaTICE), National High School of Engineers of Tunis (ENSIT), University of Tunis, Tunisia*

[2]*University of Tunis-El Manar, Tunisia*

Abstract

In this paper, we present four new algorithms called, BiBin Alter, BiBin Cons and BiBin Sim, for biclustering of binary microarray data. There are novel alternatives to extract biclusters from sparse binary datasets. Our algorithms are based on *Iterative Row* and *Column Clustering Combination* (IRCCC) and Divide and Conquer (DC) approaches, Bi Max initialization and the Cro Bin evaluation function. Applied on binary synthetic datasets, our algorithms outperform other biclustering algorithms for binary microarray data. Biclusters with different numbers of rows and columns can be detected, varying from many rows to few columns and few rows to many columns. Our algorithms allow the user to guide the search towards biclusters of specific dimensions.

Keywords: Biclustering; Binary data; Microarray data; Iteratif row column combination approach; Divide and Conquer approach; CroBin

Introduction

A DNA Microarray is a glass slide covered with a chemical product and DNA samples containingthousands of genes. By placing this glass slide under a scanner, we obtain an image in whichcolored dots that represent the expression level of genes under experimental conditions. Theobtained colored image can be coded by a matrix *M*, called *gene expression data*, or microarray data, where the i^{th} row represents the i^{th} gene, the j^{th} column represents the j^{th} condition and the cell m_{ij} represents the expression level of the i^{th} gene under the j^{th} condition. Simultaneousclustering of rows (genes) and columns (conditions) of this matrix enables to identify subsets ofgenes that have similar behavior under subsets of conditions, so we say that these genes coexpress, but behave independently under other subsets of conditions. This type of clustering is called *biclustering*. Biclustering of microarray data can be helpful to study, among others, the activityand the condition of the tissue via microarrays such as transcription factor binding, insertional mutagenesis and gene expression data. It can be helpful also to find genes involved in tumorprogression, identify the function of new genes, rank the tumors into homogenous groups' andidentify new therapeutic strategies.

The biclustering problem can be formulated as follows: Given a data matrix M, construct a group of biclusters *Bopt* associated with *M* such that:

$$f(B_{opt}) = \max_{B\in BC(M)} f(B) \tag{1.1}$$

Where *f* is an objective function measuring the quality, i.e., degree of coherence, of a group of biclusters and *BC(M)*is the set of all the possible groups of biclusters associated with *M*. Clearly, biclustering is a highly combinatorial problem with a search space size $O(2^{|I|+|J|})$. Inits general case, biclustering is NP-hard [1,2].

A natural way to visualize a group of biclusters consists in assigning a different color to each bicluster and in reordering the rows and the columns of the data matrix so that we obtain a data matrix with colored biclusters, where each bicluster represents a bicluster.

Genomic datasets often consist of large binary sparse data matrices. None of the existing biclustering algorithms for binary microarray data can handle the large number of zeros in sparsebinary matrices. In this paper, we present four new algorithms called, BiBin Alter, BiBin Cons, BiBin Sim and BiBin Max, for biclustering of binary microarray data. Our algorithms are based on *Iterative Row* and *Column Clustering Combination* (IRCCC) and *Divide and Conquer* (DC) approaches, Bi Max initialization and the CroBin evaluation function [3]. Applied on binary synthetic datasets, our algorithms outperform other biclustering algorithms for binary microarraydata.

The rest of this paper is organized as follows. In section "Related works", we present related works. Insection "Preliminaries", we introduce some preliminaries. In section "Biclustering Algorithms", we present Best BiBin our biclustering algorithms of binary microarray data. In section "Illustrative Example", we present an illustrate if example. In section"Experimental Results", we present the experimental results obtained thanks to BiBin Alter, BiBin Cons BiBin Sim and BiBin Max, on binary synthetic datasets, and we compare these results with those obtained by other biclustering algorithms of binary microarray data. Finally, in section "Conclusion" we present the conclusion.

Related Works

Biclustering algorithms of binary microarray data enable to extract useful biclusters from binary data to provide information about the distribution of patterns and intrinsic correlations [4,5]. A number of biclustering algorithms of binary microarray data have been proposed in recent years, such as the *Biclustering Bit-pattern* (BiBit) [6,7], Cmnk [8,9] , BiMax [10], *Bipartite Bron-Kerbosch* (BBK) [11], *Binary Matrix Factorization* (BMF) [12], e-CCC Biclustering [13], e-BiMotif [13],

***Corresponding author:** Haifa Ben Saber, Laboratory of Technologies of Information and Communication and Electrical Engineering (LaTICE),National High School of Engineers of Tunis (ENSIT), University of Tunis, Tunisia, Tel: 71 567 322; E-mail: bensaberhaifa1@gmail.com

Received January 19, 2015; **Accepted** February 25, 2015; **Published** March 05, 2015

Citation: Saber HB, Elloumi M (2015) Efficiently Mining Gene Expression Data via Novel Binary Biclustering Algorithms. J Proteomics Bioinform S9: 008. doi:10.4172/jpb.S9-008

BIMODULE, BIDENSE [8,9], CE-Tree [14], DeBi [15] and Maximal

Frequent Item Set [15]. Besides, there are also other approaches based on Gaussian or LatentMixture Models, BEM and BCEM [16].

In the same context, different biclustering algorithms have been adapted to deal with biclustering of binary gene expression data. However, these changes lead to more complicated userinput parameters. Besides, all the elements of every generated bicluster are set to zero in theinput matrix, introducing noise. It is interesting from the biological point of view [9] to search biclusters with small proportion of zeros especially when binary data matrix is obtained after normalization and binarization. However, most of biclustering algorithms of binary microarray data, including Cmnk and Bi Max, fail to extract pertinent biclusters on sparse binary datasets. Indeed, if we apply one of these algorithms on a typical sparse binary microarraydatasets (with thousands of columns), most of the extracted biclusters are made up only by 1's.

Preliminaries

In this section we present some preliminaries necessary for the presentation of importantformulas and relationships and the used theory.

Let *I={1,2,…,n}* be a set of indices of n genes, *J={1,2,…,m}* be a set of indices of mconditions and *M(I,J)* be a data matrix associated with *I* and *J*.

The biclustering problem of a binary microarray data boils down to a minimization of thecriterion *W(z, w, A)* defined by :

$$W(z,w,A)=\sum_{k=1}^{a}\sum_{l=1}^{m}\sum_{i\in zk}\sum_{j\in wl}\left|m_{ij-}a_{kl}\right| \qquad (3.1)$$

Where *z* is a partition of *I* into *g*clusters. It is defined by $z=(z_1,\ldots,z_g)$ or by $z=\{\overline{z}_1,\overline{z}_2,\ldots.\overline{z}_K\}$. We can introduce z as a vector that has n components, where all of z components correspond to the m rows. So in this case ∈ where z_i is a cluster number of *i*or $(z_{ik};i=1,\ldots,n;k=1,\ldots g)$. Hence, if the *i*th row is a member of*k*th cluster so, z_{ik}=1 if *i*∈ *k* and otherwise z_{ik}=0. The cardinal of the *k*th cluster of *I* is denoted by $z_k=\sum_{i=1}^{n}z_{ik}=\sum_i z_{ik}=\#\overline{z}_k$.

w is a partition of *J* into *m* clusters. It is defined by $w=(w_1,\ldots,w_g)$. We can use $w=\{\overline{w}_1,\overline{w}_1,\ldots\overline{w}_M\}$ or $w=\{w_1,w_2,\ldots,w_p\}\rightarrow(w_{jl})$ notations where w_j is a cluster number of *j* or $w=(w_{jl};j=1,\ldots,d;l=1\ldots,m)$ and w_{jl}=1 if *j* ∈ *l* and w_{jl}=0 otherwise. The cardinal of the lth cluster of w is denoted by $w_l=\sum_{j=1}^{d}w_{jl}=\sum_j w_{jl}=\#\overline{w}_l$

A=a_{kl} *is*the summary of the data matrix where *k* (resp. *l*) represents number of clusters on rows (resp. columns).We note that the bicluster *kl*is defined by the $m_{ij}$with $z_{ik}w_{jl}=1$. We note also that $\sum_{i\in\overline{z}_k}m_{ik}=\sum_{i|z_{ik}=1}m_{ij}=\sum_i z_{ik}m_{ij}$

In the binary case, we assign the most frequent score in the matrix *M*, i.e*mode*, to a_{kl} and we use the criterion $W=z_{ik}w_{jl}\sum_{i,j,k,l}\left|m_{ij}-a_{kl}\right|$. At this step, we make a minimization by fixing either *w* or *z*:

- If w is fixed, we make a minimization of :

$$W(z,A\,|\,w)=\sum_{i,k,l}z_{ik}\,|\,u_{il}-\#w_l a_{kl}\,| \qquad (3.2)$$

Where

$u_{il}=\sum_{j\in w_l}m_{ij}=\sum_j w_{jl}m_{ij},\ \sum_{i,j,k,l}z_{ik}w_{jl}\,|\,m_{ij}-a_{kl}\,|=\sum_{i,k}z_{ik}\sum_{j,l}w_{jl}\,|\,m_{ij}-a_{kl}\,|=\sum_{i,k}z_{ik}\sum_{l}|\,u_{il}-\#\overline{w}_l a_{kl}\,|$, u is a matrix of size *nxL*

- If *z* is fixed, we make a minimization of :

$$W(w,A\,|\,z)=\sum_{i,k,l}w_{jl}\,|\,v_{jl}-\#z_k a_{kl}\,| \qquad (3.3)$$

Where

$v_{kj}=\sum_{i\in z_k}m_i^j=\sum_i z_{ik}m_{ij},\ \sum_{i,j,k,l}z_{ik}w_{jl}\left|m_{ij}-a_{kl}\right|=\sum_{j,l}w_{jl}\sum_{i,k}z_{ik}\left|m_{ij}-a_{kl}\right|=\sum_{i,k}z_{ik}\sum_{l}\left|v_{kj}-\#\overline{z}_k a_{kl}\right|$, v is a matrix of size *Kxp*.

Our Biclustering Algorithms

In this section, we develop our biclustering algorithms, BiBinAlter, BiBinCons and BiBinGlob, that are based on the IRCCC approach and the CroBin evaluation function [Nadif]. It consists topermute the rows ans the columns in order to obtain homogeneous biclusters. As a preprocessingstep of all these algorithms:

(i) First, when the data matrix *M*is not a binary one, we apply a thresholding function to transform it to a binary one. To the best of our knowledge, the main thresholding functions are *discretize, normalize* and *binarize* [package biclust]. According to [BicBin, Bibit], the most adequate thresholding function to binarize microarray data is binarize.

(ii) Then, we make an initial clustering z^0 of rows and an initial clustering w^0 of columns, thanks to *k*-means algorithm [17,19].

After the initialisation, we apply an update of clusters on rows (resp. columns) by fixingcolumns (resp. rows) via a derivation of the score function to optimize the criterion of CroBin. We propose four versions to keep the best result later.

In this paper, the standardization process was applied before using the self-organizing map [19] clustering technique. The second preprocessing step consists in obtaining a binary matrix thatcan be directly used by our algorithms. This is the aim of the second step in the preprocessing. The binary values of 1 and 0 under an experimental condition c mean that a gene is expressed or not, respectively. For example, in [10], a discretization threshold was set to $\overline{e}+(\underline{e}-\overline{e})/2$, with $\underline{e}$ and $\overline{e}$ as the minimum and maximum expression values in the data matrix, respectively.

BiBin Alter

Our first biclustering algorithm, BiBin Alter, receives as input an initial position *(z^0, w^0, A^0)* where z^0 is the initial clustering of rows, w^0 is the initial clustering of columns and A0 is the summary matrix related to A^0 and w^0, and gives as output a final position *(z^{opt}; w^{opt}; A^{opt})* where z^{opt} is the final clustering of rows, w^{opt} is the final clustering of columns and A^{opt} is the summary matrix related to *z* and *w*.

By adopting BiBin Alter, we operate as follows:

First, we start from an initial position *(z^0, w^0, A^0)*

Then, at each iteration c, we compute *(z^c,w^c, A^c)* starting from *(z^{c-1}, w^{c-1}, A^{c-1})*.To do so:

First, we compute *(z^c, w^{c-1}, A')* starting from *(z^{c-1}, w^{c-1}, A^{c-1})*, by using equation 3.2 (avec latex), where *A'* is an intermediate summary matrix

Then, we compute *(z^c, w^c, A^c)*starting from *(z^c, w^{c-1}, A')*, by using equation 3.3 (avec latex).

We repeat this process until the current position *(z^c, w^c, A^c)* cannot be improved any more, i.e$(z^c,w^c, A^c)=(z^{c-1}, w^{c-1}, A^{c-1})$.

BiBin Cons

Our second biclustering algorithm, BiBin Cons, is in an improvement of the previous one,i.e. to treat the optimization criterion. Our second biclustering algorithm, BiBin Cons, receives as input an initial position (z^0, w^0, A^0), where z^0 is the initial clustering of rows, w^0 is theinitial clustering of columns and A^0 is the summary matrix related to z^0 and w^0, and gives asoutput a final position $(z^{opt}, w^{opt}, A^{opt})$, where z^{opt} is the final clustering of rows, w^{opt} is thefinal clustering of columns and A^{opt} is the summary matrix related to z and w.

By adopting BiBinCons, we operate as follows:

First, we start from an initial position (z^0, w^0, A^0)

Then, at each iteration c, we repeat computing (z^c, w^c, A^c) starting from $(z^{c-1}, w^{c-1}, A^{c-1})$.

To do so:

First, we compute (z^c, w^{c-1}, A^0) starting from $(z^{c-1}, w^{c-1}, A^{c-1})$, by using equation 3.2 (avec latex), where$A^0$is an intermediate summary matrix

Then, we compute (z^c, w^c, A^c) starting from (z^c, w^{c-1}, A^0), by using equation 3.3 (aveclatex).

We repeat this process until the current position (z^c, w^c, A^c) cannot be improved any more,i.e $(z^c, w^c, A^c)=(z^{c-1}, w^{c-1}, A^{c-1})$.

We note here that the convergence criterion is obtained following the steps 2.1 and 2.2 consecutively.

BiBin Sim

The BiBin Sim consists in an improvement of the previous version BiBinCons to treat theoptimization criterion. The algorithm proceeds as follows:

By adopting BiBin Cons, we operate as follows:

First, we start from an initial position (z^0, w^0, A^0)

Then, at each iteration c, we repeat computing (z^c, w^c, A^c) starting from $(z^{c-1}, w^{c-1}, A^{c-1})$.

To do so:

First, we compute (z^c, w^{c-1}, A') starting from $(z^{c-1}, w^{c-1}, A^{c-1})$, by using equation 3.2(avec latex), where A^0 is an intermediate summary matrix

Then, we compute (z^c, w^c, A^c) starting from (z^c, w^{c-1}, A'), by using equation 3.3 (aveclatex).

We repeat this process until the current position (z^c, w^c, A^c) cannot be improved any more, i.e $(z^c, w^c, A^c)=(z^{c-1}, w^{c-1}, A^{c-1})$.

Finally, we choose the best convergence criterion obtained according the application to therows and columns.

BiBin Max

The BiBin Sim consists in an improvement of the previous version BiBin Sim and Bimax algorithm to treat the optimization criterion. The algorithm proceeds as follows:

By adopting BiBin Max, we operate as follows:

First, we start from an initial position (z^0, w^0, A) where A consider the initialization with Bi Max algorithm.

Then, at each iteration c, we repeat computing (z^c, w^c, A) starting from (z^{c-1}, w^{c-1}, A). To do so:

First, we compute (z^c, w^{c-1}, A) starting *from* (z^{c-1}, w^{c-1}, A), by using equation 3.2.

Then, we compute (z^c, w^c, A) starting from (z^c, w^{c-1}, A), by using equation 3.3.

We repeat this process until the current position (z^c, w^c, A) cannot be improved any more, i.e$(z^c, w^c, A)=(z^{c-1}, w^{c-1}, A)$.

Finally, we choose the best convergence criterion obtained according the application to therows and columns.

Illustrative Example

We present in this section steps to perform the biclustering on binary datasets. Our four versions allow to reorder the rows and the columns of the data matrix in both dimensions to obtain homogeneous biclusters. The algorithm minimizes the difference between the initial matrixaccording the two ways and the ideal matrix.

To illustrate our algorithm, we propose to run it on a simple example. Let M be a 9×11matrix of binary data to perform biclustering. The initialization is to group the rows and columns with the dynamic clusters method. After initialization, we compute z and w matrixwhose elements determine the membership of rows or column in horizontal or vertical clusters,respectively. Then, we reorganize the binary matrix. After that, we compute the summarymatrix is obtained from z and w and the bicluster kl is defined by the x_{ij}'s with $z_{ik}w_{jl}=1$.The summary matrix is presented by the major value a_{kl}: This parameter presents the degreeof homogeneity via summary matrix. The initial data matrix M is resumed by a simpler data matrix a_{kl} having the same structure. Finally, we update clusters and application of the criterionon both dimensions (Figure 1):

- Data matrix reorganized according a partition of rows: Partition z of I in K clusters

- Data matrix reorganized according a partition of rows and columns: Partition w of J in M clusters

These results seem to be promising to study our method on real data sets and compare itwith other biclustering algorithms.

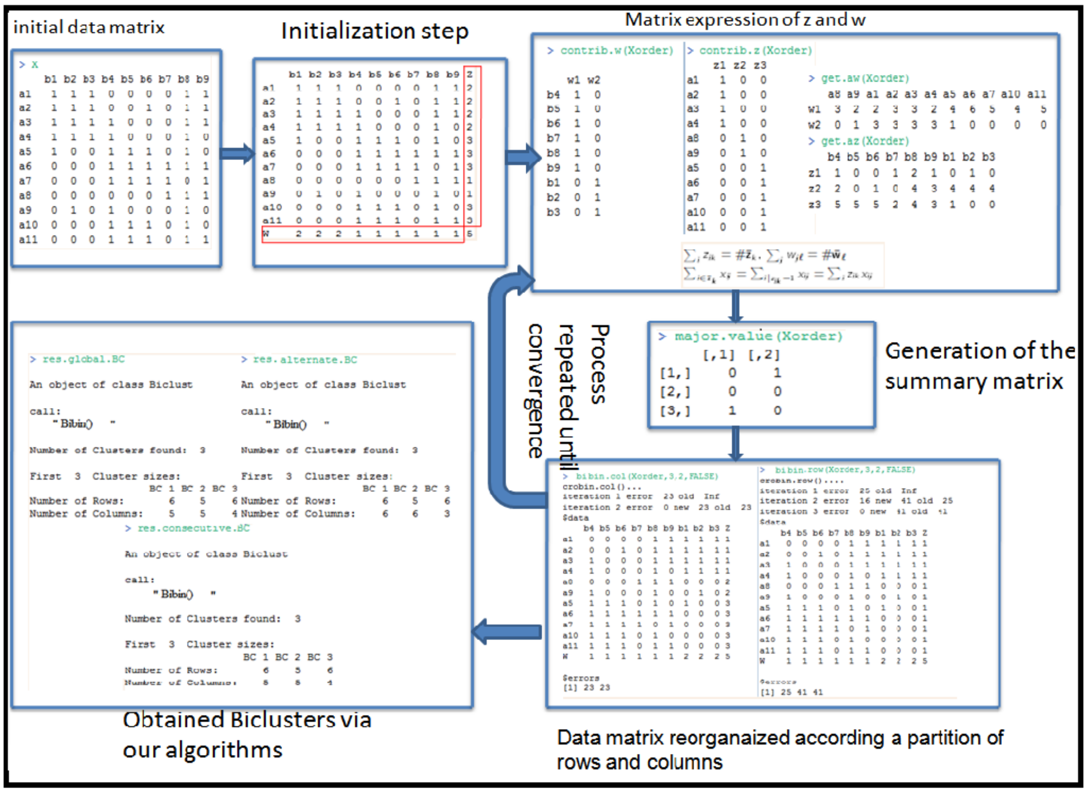
The same process is applied for the BiBin Max algorithm by using the initialization for BiMax.In fact, we had considered BiMax parameters for akl to the constant values. Our result isequivalent with which obtained by BiMax.

Experimental Results

Synthetic data

For illustrating the structures of the Bibin Ater, Bibin Cons, BibinSim, BiBin Max algorithms, we have designed some simulated data sets from the bicluster mixture model for comparing theperformance of the algorithms. It should be mentioned that in following process, we are goingto study the algorithms' behaviours in which there are variations in the degree of homogeneityand the proportions of the clusters. To do this, we have applied synthetic data sets accordingthese characteristics (Table 1):

The CroBin algorithm which is seemed fast and gives good results when the biclusters have thesame proportions and degrees of homogeneity similar except that you set the number of groupson the


initial data matrix
Initialization step
Matrix expression of z and w
Process repeated until convergence
Generation of the summary matrix
Obtained Biclusters via our algorithms
Data matrix reorganaized according a partition of rows and columns

```
> X
    b1 b2 b3 b4 b5 b6 b7 b8 b9
a1   1  1  1  0  0  0  0  1  1
a2   1  1  1  0  0  1  0  1  1
a3   1  1  1  1  0  0  0  1  1
a4   1  1  1  1  0  0  0  1  0
a5   1  0  0  1  1  1  0  1  0
a6   0  0  0  1  1  1  1  1  1
a7   0  0  0  1  1  1  1  0  1
a8   0  0  0  0  0  0  1  1  1
a9   0  1  0  1  0  0  0  1  0
a10  0  0  0  1  1  1  0  1  0
a11  0  0  0  1  1  1  0  1  1
```

```
    b1 b2 b3 b4 b5 b6 b7 b8 b9 Z
a1   1  1  1  0  0  0  0  1  1 2
a2   1  1  1  0  0  1  0  1  1 2
a3   1  1  1  1  0  0  0  1  1 2
a4   1  1  1  1  0  0  0  1  0 2
a5   1  0  0  1  1  1  0  1  0 3
a6   0  0  0  1  1  1  1  1  1 3
a7   0  0  0  1  1  1  1  0  1 3
a8   0  0  0  0  0  0  1  1  1 1
a9   0  1  0  1  0  0  0  1  0 1
a10  0  0  0  1  1  1  0  1  0 3
a11  0  0  0  1  1  1  0  1  1 3
W    2  2  2  1  1  1  1  1  1 5
```

```
> contrib.w(Xorder)        > contrib.z(Xorder)
                               z1 z2 z3
   w1 w2                   a1   1  0  0
b4  1  0                   a2   1  0  0
b5  1  0                   a3   1  0  0
b6  1  0                   a4   1  0  0
b7  1  0                   a8   0  1  0
b8  1  0                   a9   0  1  0
b9  1  0                   a5   0  0  1
b1  0  1                   a6   0  0  1
b2  0  1                   a7   0  0  1
b3  0  1                   a10  0  0  1
                           a11  0  0  1
```

```
> get.aw(Xorder)
   a8 a9 a1 a2 a3 a4 a5 a6 a7 a10 a11
w1  3  2  2  3  3  2  4  6  5   4   5
w2  0  1  3  3  3  3  1  0  0   0   0
> get.az(Xorder)
   b4 b5 b6 b7 b8 b9 b1 b2 b3
z1  1  0  0  1  2  1  0  1  0
z2  2  0  1  0  4  3  4  4  4
z3  5  5  5  2  4  3  1  0  0
```

$\sum_i z_{ik} = \#\bar{z}_k,\ \sum_j w_{j\ell} = \#\bar{w}_\ell$

$\sum_{i \in z_k} x_{ij} = \sum_{i | z_{ik}=1} x_{ij} = \sum_i z_{ik} x_{ij}$

```
> major.value(Xorder)
     [,1] [,2]
[1,]    0    1
[2,]    0    0
[3,]    1    0
```

```
> bibin.col(Xorder,3,2,FALSE)
crobin.col()...
iteration 1 error  23 old  Inf
iteration 2 error  0 new  23 old  23
$data
    b4 b5 b6 b7 b8 b9 b1 b2 b3 Z
a1   0  0  0  0  1  1  1  1  1 1
a2   0  0  1  0  1  1  1  1  1 1
a3   1  0  0  0  1  1  1  1  1 1
a4   1  0  0  0  1  0  1  1  1 1
a8   0  0  0  1  1  1  0  0  0 2
a9   1  0  0  0  1  0  0  1  0 2
a5   1  1  1  0  1  0  1  0  0 3
a6   1  1  1  1  1  1  0  0  0 3
a7   1  1  1  1  0  1  0  0  0 3
a10  1  1  1  0  1  0  0  0  0 3
a11  1  1  1  0  1  1  0  0  0 3
W    1  1  1  1  1  1  2  2  2 5

$errors
[1] 23 23
```

```
> bibin.row(Xorder,3,2,FALSE)
crobin.row()....
iteration 1 error  25 old  Inf
iteration 2 error  16 new  41 old  25
iteration 3 error  0 new  41 old  41
$data
    b4 b5 b6 b7 b8 b9 b1 b2 b3 Z
a1   0  0  0  0  1  1  1  1  1 1
a2   0  0  1  0  1  1  1  1  1 1
a3   1  0  0  0  1  1  1  1  1 1
a4   1  0  0  0  1  0  1  1  1 1
a8   0  0  0  1  1  1  0  0  0 1
a9   1  0  0  0  1  0  0  1  0 1
a5   1  1  1  0  1  0  1  0  0 1
a6   1  1  1  1  1  1  0  0  0 1
a7   1  1  1  1  0  1  0  0  0 1
a10  1  1  1  0  1  0  0  0  0 1
a11  1  1  1  0  1  1  0  0  0 1
W    1  1  1  1  1  1  2  2  2 5

$errors
[1] 25 41 41
```

```
> res.global.BC                      > res.alternate.BC

An object of class Biclust           An object of class Biclust

call:                                call:
   " Bibin()   "                        " Bibin()   "

Number of Clusters found:  3         Number of Clusters found:  3

First  3  Cluster sizes:             First  3  Cluster sizes:
                      BC 1 BC 2 BC 3                       BC 1 BC 2 BC 3
Number of Rows:          6    5    6 Number of Rows:          6    5    6
Number of Columns:       5    5    4 Number of Columns:       6    6    3
```

```
> res.consecutive.BC

An object of class Biclust

call:
   " Bibin()   "

Number of Clusters found:  3

First  3  Cluster sizes:
                      BC 1 BC 2 BC 3
Number of Rows:          6    5    6
Number of Columns:       5    5    4
```

Figure 1: Illustrative Example.

faParameters		
Characteristics of the data	Number of biclusters	g=3 and m=2
	Model	Latent bicluster Bernoulli Mixture
Situations	Pattterns of overlapping	Well separated + 5% Fairly separated : ++ 15% Poorly Separated +++ 25%
	Sizes of data	Small : 50 × 30 Medium :100 × 60 Large : 200 × 120

Table 1: Characteristics and situations of the data.

rows and columns. However, it seems to be bad when the proportions of partitions aredramatically different which leads to think that CroBin assumes equal proportions of clusters. Ituses the algorithm of dynamic clusters [17,19] to optimize the criteria. This algorithm rearrangesthe rows and columns of the data matrix along the two sheets of the rows and columns of homogeneous biclusters. The algorithm minimizes the difference between the initial matrixstructured and the ideal matrix according scores.

The summarize of the most important points obtained from these simulations are as follows:Obviously, we can interpret the reasonable results according to the model underlying the datastructure: In this research, we proves that if the proportions of the components are considered equal, Bibin Alter, Bib Cons and BiBin Sim give good results. The convergence has a fast progress:most of the time. The BiBin Max is faster and gives us interesting results. The obtained bicluster is very clear since extracted biclusters containing a majority number of '1' and very illustrative to help the biologist to extract knowledge. According to our implementations, we note first thatthe choice and application of a given criterion is not always obvious or easy to find.

The best bicluster is maximally dense with ones, but the number of ones can in principle vary.Moreover, for each input matrix, the number of ones in the maximal bicluster will be different. The user would probably prefer to indicate in advance the proportion of ones that should be in a sub-matrix before it is called a bicluster, even though the sub-matrix has maximum. In order to find biclusters that the user would consider dense enough, we introduce a functionto calculateproportion of ones that is largest. We note that a bicluster which has a proportion of zeros can contain all the interesting information. To further enrich this bicluster for ones, i.e., to find a more dense bicluster that meets the constraint, we run the algorithm again on the maximal bicluster (Figure 2).

By considering the error percentages of algorithms and the size of the matrices, we note that BiBin Max gives good results compared to other proposed algorithms. In fact, according to our study, we find that Bibin Cons Divergent when the matrix becomes more large. We remark also that the error rates are proportional according the overlap rate. Besides, we show that BiBin Max, BiBin Cons BiBin Alter and look the same in the different case studies. Thus, we consider forthe rest of this work only the 4th version to compare it with other algorithms mentioned inliterature and designated for binary data.

Real dataset

The Yeast dataset consist of a data matrix composed by 2884 genes (rows) and 17 experimental conditions (columns). We use to gather statistical information about this data matrix to decide which parameters are suitable for our purposes.

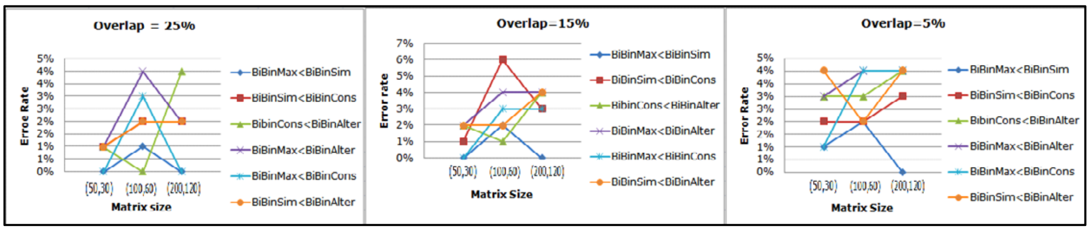


Figure 2: Error Rate of compared algorithms on binary matrices with different sizes.

	BibinCons	BibinAlter	BibinMax	BibinSim
Avg. MSR	234.87	230.88	218.54	208.66
Avg. Size	10301.71	10502.1	10510.8	11085.44
Avg. genes	1095.43	1062.2	1102.84	1118.41
Avg. cond	9.29	9.3	9.31	9.45
Max Size	14828	14899	15613	15795

Table 2: Performance of our four algorithms.

	Avg. Residue	Avg. volume	Avg. gene numb.	Avg cond. Number	Time
CC	204.293	1576.98	167	12	12 min
Floc	187.543	1825.78	195	12.8	6.7 min
BiModule	330.65	1036.86	240.97	5.08	5.08 min
BiDENS	191.38	191.38	248.97	13.1	13.1 min
SEBI	205.18	209.92	13.61	15.25	-
BLN	70.02	166.84	15.34	10.88	-
BLNBiBinMax	70	165.8	15.2	10.5	8.12 min

Table 3: Performance comparison between algorithms.

Table 2 shows that the biclusters found by Bibin Sim is characterized by a slightly lowersquared residue and a higher bicluster size than those by Bibin Cons and Bibin Max on both yeast dataset and human dataset. However when comparing Bibin Max with BiBin Cons, we find the biclusters found by Bibin Max better quality than those found by Bibin Cons and Bibin Alter. The biclusters produced by BiBin Max is larger and has less average residue than biclusters of any of the other four algorithms. As the range that a level represents decreases the residue expected todecreases and would reach zero if the range was zero. The 100 biclusters reported for BiBin Max may overlap by more than %25 with other biclusters. If we allow overlapping by less than %25 only then only 30 biclusters could be reported. To decrease the residue of biclusters produced by BiBin Max we need to use discretization levels greater than 7 which is the number of levels recommended in [12] which will result in longer runtime and smaller bicluster size.

Table 2 compares the performance of our four algorithms, and gives the average of meansquared residue, the average number of genes and conditions, the average size and maximal size of the found biclusters.

To compare the performance of the best proposed algorithm BiBin Max with other algorithms, a criteria the coverage is defined as the total number of cells in microarray data matrices covered by the found biclusters.

The criteria used to measure the quality of biclusters are the maximum number of genes, theminimum mean squared residue, and the maximum number of conditions, in this order.

The biclusters selected present a small MSR value, i.e., it exists a great coherence acrossboth genes and conditions. We obtain biclusters with high number of genes as well, being 17the maximum value. In the

Bicluster	Genes	Conditions	Residue	Row Variance
1	79	17	205.44	711.08
8	101	16	221.12	685.33
12	621	10	200.11	1634.32
21	1156	12	221.42	1385.08
32	543	15	199.11	986.09
44	325	13	2231.04	999.55
53	1215	16	281.82	778.73
69	87	8	209.33	1085.22
81	1224	9	201.77	943.45
88	1022	8	203.89	911.75

Table 4: Information of biclusters found on yeast dataset.

Yeast data set this number of genes and the MSR value vary withthe parameters, i.e., when the distance threshold grows and the minimum number of conditionsdecreases (less restrictive experimental conditions), then the number of genes increases substantially.

We compare our 100 best biclusters and their average values with those obtained by thealgorithms cited in the literature and BiBin Max algorithm. BiBin Max algorithm obtains better results with regard to the MSR value than the other two algorithms. The averaged volume, i.e., the number of genes multiplies by the number of conditions, and the averages of conditions in biclusters are lower. The average number of genes is 15.2. The most interesting property of biclusters found by BiBin Max is that it provides biclusters with very low mean squared residue in comparison to the other techniques while maintaining the number of genes between 14 and 17 (Table 3).

First column (algorithm), second column (average mean squared residue), third column (average volume), fourth column (average number of genes), fifth column (average number of genes) and sixth column (average number of conditions).

Table 4 shows the number of genes and conditions, the mean squared residue and the row variance of ten biclusters out of the one hundred biclusters found on the yeast dataset.

In the first column, the identifier of each bicluster is reported. The second and third columnsreport the number of rows (genes) and of columns (conditions) of the bicluster, respectively, the fourth column reports the mean squared residues, and the last column reports the row variance of the biclusters.

Coverage measurement: To evaluate the performance of BiBin Max, we compute the total number of cells in thedataset that are covered by the biclusters are used. Our 100 biclusters selected 20 cover 51.76% cells of the initial dataset, while this coverage is 13.36% for BiBin Sim. The poor coverage of BiBin Sim can be explained by the transformationof the summary matrix at each iteration. BiBinMax can extract biclusters offering a much bettercoverage.

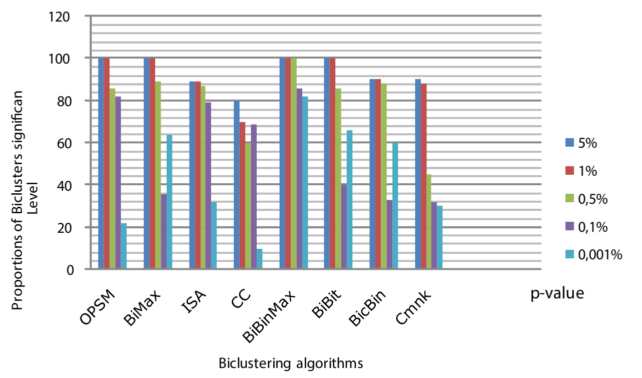


Figure 3: Proportions of Biclusters significantly enriched by GO annotations on yeast cell-cycle dataset.

Cluster No.	No. of genes	Process Function	Component
16	Lipid transport (n =23, p=0.00013)	Oxidoreductase activity (n=12, p=0.00376)	membrane (n=18, p=0.0064
56	Cell organization and biogenesis (n=31, p= 0.0046)	Protein transporter activity (n=5, p=0.0035)	Nucleus (n=25, p=0.0043)
81	Cellular process (n=37, p=0.0023)	tRNA methyltransferase activity (n=14, p=0.0012)	Cytosolic small ribosomal subunit (n= 11, p=0.0065)

Table 5: Signi_cant GO terms (process, function, component) on yeast cell-cycle dataset.

Biological relevance: In order to evaluate the biological relevance of BiBinMax, we use again the p-values andapply the web-tool FuncAssociate. The results of BiBinMax are comparedagainst reported scores of CC, ISA, Bimax, OPSM and BiBinMax. We note that BiBinMaxperforms well for all p-values compared to CC, ISA and OPSM. Also, BiBinMax performs wellfor four cases of pvalue (p-value=5%, p-value=1%, p-value=0.5% and ≍ pvalue=0.1%) overfive compared to Bimax. Best results are obtained by BiBinMax and Bimax.

Figure 3 shows, for each significant score p (p=5%, 1%, 0.5%, 0.1% and 0.001%) andfor each compared algorithm, the percentage of the statistically significant biclusters extractedby the algorithm with the indicated p-value. We observe that BiBinMax outperforms the otheralgorithms on this dataset. 100% of discovered biclusters of BiBinMax are statistically significantwith p=5%, 1%, 0.5% and 0.1%. Even with $p \leq 0.001\%$, 89% of discovered biclusters of BiBinMaxare statistically significant against 51% for BiBinSim and BiBinCons and 64% for Bimax. With p=0.5%, 0.1%, no competing algorithm can offer 100% statistically significant biclusters.

Analysis of biological annotation enrichment of biclusters: In order to identify the biological annotations for the biclusters, we use GOTerm Finder(http://db.yeastgenome.org/cgi-bin/GO/goTermFinder). Table 5 lists the significant sharedGO terms which are used to describe genes in each bicluster for the process, function and componentontology. Here, only shows the terms whose p-values are smaller than 0.01.

This table lists the significant shared GO terms (or parent of GO terms) used to describe theset of genes in each bicluster for the process, function and component ontology. For example forcluster C16, we find that the genes are mainly involved in lipid transport. The tuple (n=23, p= 0.00013) means that out of 96 genes in cluster C16, 23 genes belong to lipid transport process,and the statistical significance is given by the p-value of 0.00013. Those results mean that the proposed BiBinMax biclustering approach can find biologically meaningful clusters.

We apply the yeast genome gene ontology term finder on each discovered biclusters to evaluate their biological significance in terms of associated biological processes, molecular functions andcellular components respectively. The GO terms are displayed in the decreasing order of significance. For the bicluster labelled B1, the genes RFA1, POL12, POL30, CDC9, MSH6, RAD27,CDC45, RFA2, and CDC21 are together involved in the process of DNA-dependent DNA replication,DNA replication and DNA metabolic process. Each GO term is associated with a tuple, for example DNA-dependent DNA replication (9, 1.40e-11) indicates 9 out of the total 16 genesof B1 belong to this process and their statistical significance is 1.40e-11 i.e. p-value. Also fromthe table it is clear that the biclusters extracted are distinct along each category. Existence of biclusters comprising a significant proportion of those genes that are considered similar biologicallyis proof that a specific biclustering technique produces biologically relevant results. This showsthat our algorithm is capable of identifying a broader range of biologically significant biclusters.

Conclusion

In this paper, we presented our method of binary biclustering considered to be relevant by theexpert starting from the data resulting from the microarrays. The suggested method generates the best bicluster. To achieve this purpose, we selected data to which we applied a thresholding to release the binary data, and four versions relatives to our package. These four versions exploitingthe criterion to search biclusters. Our four proposals have been implemented and evaluated onsynthetic and real datasets. According to our implementations, we note first that the choice and application of a given criterion is not always obvious or easy to find. Enjoying the benefits of our algorithms, we proposed a methodology for the identification of homogeneous biclusters. The first results are very encouraging and persuade us of the obviousinterest of such an approach.

References

1. Cheng KO, Law NF, Siu WC, Liew AWC (2008) Identification of coherent patterns in gene expression data using an efficient biclustering algorithm and parallel coordinate visualization. BMC Bioinformatics.
2. Madeira SC, Oliveira AL (2004) Biclustering algorithms for biological data analysis: a survey. IEEE/ACM Trans Comput Biol Bioinform 1: 24-45.
3. Govaert G (1983) La classification croisee. Modulad.
4. Serin A (2011 Biclustering analysis for large scale data.
5. Roy S, Bhattacharyya DK, Kalita JK (2013) Cobi: Pattern based coregulated biclustering of gene expression data. Pattern Recognition Letters.
6. Rodriguez-Baena DS, Perez-Pulido AJ, Aguilar-Ruiz JS (2011) A biclustering algorithm for extracting bit-patterns from binary datasets. Bioinformatics 27: 2738-2745.
7. Aguilar-Ruiz JS (2005) Shifting and scaling patterns from gene expression data. Bioinformatics 21: 3840-3845.
8. Koyuturk M, Szpankowski W, Grama A (2004) Biclustering gene-feature matrices for statistically significant dense patterns. In 2004 IEEE Computational Systems Bioinformatics Conference (CSB'04).
9. Koyuturk M (2012) Using protein interaction networks to understand complex diseases. Computer 45: 31-38.
10. Preli¢ A, Bleuler S, Zimmermann P, Wille A, Bühlmann P, et al. (2006) A systematic comparison and evaluation of biclustering methods for gene expression data. Bioinformatics 22: 1122-1129.
11. Voggenreiter O, Bleuler S, Gruissem W (2012) Exact biclustering algorithm for

the analysis of large gene expression data sets. Eighth International Society for Computational Biology (ISCB) Student Council Symposium Long Beach, CA, USA.

12. Zhang ZY, Li T, Ding C, Ren XW, Zhang XS (2010) Binary matrix factorization for analyzing gene expression data. Data Mining and Knowledge Discovery 20: 28-52.
13. Gonçalves JP, Madeira SC (2010) e-bimotif: Combining sequence alignment and biclustering to unravel structured motifs. In IWPACBB 74: 181-119.
14. Chen JR, Chang YI (2009) A Condition-Enumeration Tree method for mining biclusters from DNA microarray data sets. Biosystems 97: 44-59.
15. Serin A, Vingron M (2011) DeBi: Discovering Differentially Expressed Biclusters using a Frequent Itemset Approach. Algorithms Mol Biol 6: 18.
16. Charrad M (2010) A gnrique -analysis approach for crossing content and use of websites by methods of bipartitionnement. Paris and ENSI, University of Manouba.
17. da Silva MAS, Monteiro AMV, Câmara G (2013) Som-code: Design patterns and generic programming in the implementation of self organizing maps.
18. Benabdeslem K, Allab K (2013) Bi-clustering continuous data with self-organizing map. Neural Computing and Applications 22: 1551-1562.
19. Benabdeslem K, Allab K, Aussem A (2012) Automatic co-clustering approach based topological maps. RNTI.

This article was originally published in a special issue, **Computational Intelligence in Bioinformatics** handled by Editor(s). Dr. Jean-Christophe Nebel, Kingston University, London, UK