Efficiently Mining Gene Expression Data via Novel Binary Biclustering Algorithms

Haifa Ben Saber*1 and Mourad Elloumi1,2

1Laboratory of Technologies of Information and Communication and Electrical Engineering (LaTICE), National High School of Engineers of Tunis (ENSIT), University of Tunis, Tunisia
2University 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, BiMax initialization and the CroBin 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; Iterative 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 containing thousands of genes. By placing this glass slide under a scanner, we obtain an image in which colored dots that represent the expression level of genes under experimental conditions. The obtained 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. Simultaneous clustering of rows (genes) and columns (conditions) of this matrix enables to identify subsets of genes 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 activity and 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 tumorigenesis, identify the function of new genes, rank the tumors into homogenous groups and identify new therapeutic strategies.

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

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

(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 $O(2^{n \times m})$. In its 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 sparse binary 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, BiMax initialization and the CroBin evaluation function. Applied on binary synthetic datasets, our algorithms outperform other biclustering algorithms for binary microarray data.

The rest of this paper is organized as follows. In section "Related works", we present related works. In section "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],...
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 user input parameters. Besides, all the elements of every generated bicluster are set to zero in the input 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 Cmmk 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 microarray datasets (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 important formulas 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 m conditions 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 the criterion \( W(z, w, A)\) defined by:

\[
W(z, w, A) = \sum_{i} \sum_{j} \sum_{k} \sum_{l} |m_{ijkl} - a_{ijkl}| \tag{3.1}
\]

Where \( z \) is a partition of \( I\) into clusters. It is defined by \( z(\varepsilon_{1},...,\varepsilon_{g})\) or by \( z = \{\varepsilon_{1},...,\varepsilon_{g}\}\). We can introduce \( \varepsilon\) as a vector that has \( n\) components, where all of \( x\) components correspond to the \( m\) rows. So in this case \( \varepsilon_{i}\) is a cluster number of ior \( i = 1,...,n, k = 1,...g\) . Hence, if the \( i\) row is a member of \( k\) cluster, \( z_{ik}=0\) if \( \varepsilon_{i}\) is fixed, we make a minimization of:

\[
W(x, A) = \sum_{i} y_{ij} |v_{ij} - \# z_{ij}a_{ij}| \tag{3.3}
\]

Where \( v_{ij} = \sum_{g_{c}} m'_{ij} = \sum_{g_{c}} m_{ijkl} - \sum_{ij} m_{ijkl} a_{ijkl} \) or \( v\) is a matrix of size \( Kxp\).

Our Biclustering Algorithms

In this section, we develop the biclustering algorithms, BiBin Alter, BiBinCons and BiBinGlob, that are based on the IRCCC approach and the CroBin evaluation function [Nadif]. It consists to permute the rows and the columns in order to obtain homogeneous biclusters. As a preprocessing step 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 discrete, normalize and binarize [package biclust]. According to [BicBin, Bibt], the most adequate thresholding function to binarize microarray data is binarize.

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

After the initialisation, we apply an update of clusters on rows (resp. columns) by fixing columns (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 that can 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 mean that a gene is expressed or not, respectively. For example, in [10], a discretization threshold was set to \( \frac{E - \overline{E}}{2E} \) with \( E \) 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 \((z^{c}, w^{c}, A^{c})\) where \( z^{c}\) is the initial clustering of rows, \( w^{c}\) is the initial clustering of columns and \( A^{c}\) is the summary of the data matrix where \( k \) (resp. \( l\)) represents number of rows (resp. columns). We note that the bicluster \( A^{c}\) is defined by the \( m_{ijkl}\) with \( z_{ik}w_{jl}=1\). We note also that \( \sum_{j} m_{ijkl}a_{ijkl}\) and \( \sum_{i} m_{ijkl}a_{ijkl}\).

In the binary case, we assign the most frequent score in the matrix \( M\), Le mode, to \( a_{ij}\), and we use the criterion \( W = \sum_{j} w_{ij} \sum_{j} m_{ijkl} - a_{ijkl}\). At this step, we make a minimization by fixing either \( w\) or \( z\):

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

\[
W(z, A | w) = \sum_{i} z_{ik} |a_{ik} - \# w_{ij}a_{ij}| \tag{3.2}
\]

Where

- If \( z\) is fixed, we make a minimization of:

\[
W(x, A | z) = \sum_{i} w_{ij} |v_{ij} - \# z_{ij}a_{ij}| \tag{3.3}
\]

Where

\[
v_{ij} = \sum_{g_{c}} m'_{ij} = \sum_{g_{c}} m_{ijkl} - \sum_{ij} m_{ijkl} a_{ijkl} \]

\( v\) is a matrix of size \( Kxp\).
be improved any more, i.e \( (z_c', w_c', A') = (z_c - 1, w_c - 1, A_c - 1) \).

**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 \( (z_c, w_c, A_c) \), where \( z_c \) is the initial clustering of rows, \( w_c \) is the initial clustering of columns and \( A_c \) is the summary matrix related to \( z \) and \( w \), and gives as output a final position \( (z_c'', w_c'', A_c'') \), where \( z_c'' \) is the final clustering of rows, \( w_c'' \) is the final clustering of columns and \( A_c'' \) is the summary matrix related to \( z \) and \( w \).

By adopting BiBin Cons, we operate as follows:

First, we start from an initial position \( (z_c', w_c', A_c') \)

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 \( (z_c', w_c', A_c') \) starting from \( (z_c - 1, w_c - 1, A_c - 1) \), by using equation 3.2 (avec latex), where \( A_1 \) is an intermediate summary matrix.

Then, we compute \( (z_c', w_c', A_c') \) starting from \( (z_c - 1, w_c - 1, A_c - 1) \), 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 \( (z_c', w_c', A_c') = (z_c - 1, w_c - 1, A_c - 1) \).

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 matrix according the two ways and the ideal matrix.

To illustrate our algorithm, we propose to run it on a simple example. Let \( M \) be a \( 9 \times 11 \) matrix 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 \( w \) matrix whose elements determine the membership of rows or columns in horizontal or vertical clusters, respectively. Then, we reorganize the binary matrix. After that, we compute the summary matrix is obtained from \( z \) and \( w \) and the bicluster \( k_l \) is defined by the \( x_{ij} \) with \( z_{iwj} = 1 \). The summary matrix is presented by the major value \( a_z \). This parameter presents the degree of homogeneity via summary matrix. The initial data matrix \( M \) is resumed by a simpler data matrix \( a_z \) having the same structure. Finally, we update clusters and application of the criterion to 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 it with 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 \( a_{kl} \) to the constant values. Our result is equivalent 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 the performance of the algorithms. It should be mentioned that in following process, we are going to study the algorithms’ behaviours in which there are variations in the degree of homogeneity and the proportions of the clusters. To do this, we have applied synthetic data sets according these characteristics (Table 1):

- The CroBin algorithm which is seemed fast and gives good results when the biclusters have the same proportions and degrees of homogeneity similar except that you set the number of groups on the
rows and columns. However, it seems to be bad when the proportions of partitions are dramatically different which leads to think that CroBin assumes equal proportions of clusters. It uses the algorithm of dynamic clusters [17,19] to optimize the criteria. This algorithm rearranges the 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 matrix structure 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 data structure. In this research, we prove 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 that the 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 function to calculate proportion 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 for the rest of this work only the 4th version to compare it with other algorithms mentioned in literature 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.
The biclusters selected present a small MSR value, i.e., it exists a great coherence across both genes and conditions. We obtain biclusters with high number of genes as well, being 17 the maximum value. In the first column, the identifier of each bicluster is reported. The second column (average mean squared residue), third column (average volume), fourth column (average number of genes) and fifth column (average number of conditions) are then reported. To decrease the residue of biclusters produced by BiBin Max we need to use discretization levels which will result in longer runtime and smaller bicluster size. 

Table 3: Performance comparison between algorithms.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CC</td>
<td>204.29</td>
<td>167.96</td>
<td>12</td>
<td>12</td>
<td>12 min</td>
</tr>
<tr>
<td>Floc</td>
<td>187.54</td>
<td>1825.78</td>
<td>195</td>
<td>12.8</td>
<td>6.7 min</td>
</tr>
<tr>
<td>BiModule</td>
<td>330.65</td>
<td>1036.86</td>
<td>240.97</td>
<td>5.08</td>
<td>5.08 min</td>
</tr>
<tr>
<td>BiDENS</td>
<td>191.38</td>
<td>191.38</td>
<td>248.97</td>
<td>13.1</td>
<td>13.1 min</td>
</tr>
<tr>
<td>SEBI</td>
<td>205.18</td>
<td>209.92</td>
<td>13.61</td>
<td>15.25</td>
<td>-</td>
</tr>
<tr>
<td>BLN</td>
<td>70.02</td>
<td>166.84</td>
<td>15.34</td>
<td>10.88</td>
<td>-</td>
</tr>
<tr>
<td>BLN/BiMax</td>
<td>70</td>
<td>165.8</td>
<td>15.2</td>
<td>10.5</td>
<td>8.12 min</td>
</tr>
</tbody>
</table>

Table 4: Information of biclusters found on yeast dataset.

<table>
<thead>
<tr>
<th>Bicluster</th>
<th>Genes</th>
<th>Conditions</th>
<th>Residue</th>
<th>Row Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>79</td>
<td>17</td>
<td>205.44</td>
<td>711.08</td>
</tr>
<tr>
<td>8</td>
<td>101</td>
<td>16</td>
<td>221.12</td>
<td>685.33</td>
</tr>
<tr>
<td>12</td>
<td>621</td>
<td>10</td>
<td>200.11</td>
<td>1634.32</td>
</tr>
<tr>
<td>21</td>
<td>1156</td>
<td>12</td>
<td>221.42</td>
<td>1385.08</td>
</tr>
<tr>
<td>32</td>
<td>543</td>
<td>15</td>
<td>199.11</td>
<td>986.09</td>
</tr>
<tr>
<td>44</td>
<td>325</td>
<td>13</td>
<td>2231.04</td>
<td>999.55</td>
</tr>
<tr>
<td>53</td>
<td>1215</td>
<td>16</td>
<td>281.82</td>
<td>778.73</td>
</tr>
<tr>
<td>69</td>
<td>87</td>
<td>8</td>
<td>209.33</td>
<td>1085.22</td>
</tr>
<tr>
<td>81</td>
<td>1224</td>
<td>9</td>
<td>201.77</td>
<td>943.46</td>
</tr>
<tr>
<td>88</td>
<td>1022</td>
<td>8</td>
<td>203.89</td>
<td>911.75</td>
</tr>
</tbody>
</table>

Table 2: Performance of our four algorithms.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Avg. MSR</th>
<th>Avg. Size</th>
<th>Avg. genes</th>
<th>Avg. cond.</th>
<th>Max Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>BibinCons</td>
<td>234.87</td>
<td>10301.71</td>
<td>1095.43</td>
<td>9.29</td>
<td>14828</td>
</tr>
<tr>
<td>BibinAlter</td>
<td>230.88</td>
<td>10502.1</td>
<td>1062.2</td>
<td>9.3</td>
<td>14899</td>
</tr>
<tr>
<td>BibinMax</td>
<td>218.54</td>
<td>10510.8</td>
<td>1102.84</td>
<td>9.31</td>
<td>15613</td>
</tr>
<tr>
<td>BibinSim</td>
<td>208.66</td>
<td>11085.44</td>
<td>1118.41</td>
<td>9.45</td>
<td>15795</td>
</tr>
</tbody>
</table>

Yeast data set this number of genes and the MSR value vary with the parameters, i.e., when the distance threshold grows and the minimum number of conditions decreases (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 the algorithms 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 conditions) 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 columns report 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 the dataset 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 transformation of the summary matrix at each iteration. BiBinMax can extract biclusters offering a much better coverage.
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 and cellular components respectively. The GO terms are displayed in the decreasing order of significance. For the bicluster labeled 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 (9.1.40c-11) indicates 9 out of the total 16 genes of B1 belong to this process and their statistical significance is 1.40c-11 i.e. p-value. Also from the 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 biologically is proof that a specific biclustering technique produces biologically relevant results. This shows that 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 the expert 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 of each package. These four versions exploit the criterion to search biclusters. Our four proposals have been implemented and evaluated on synthetic 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 propose a methodology for the identification of homogeneous biclusters. The first results are very encouraging and persuade us of the obvious interest of such an approach.

**References**

the analysis of large gene expression data sets. Eighth International Society
for Computational Biology (ISCB) Student Council Symposium Long Beach,
CA, USA.
for analyzing gene expression data. Data Mining and Knowledge Discovery
20: 28-52.
and biclustering to unravel structured motifs. In IWPACBB 74: 181-119.
biclusters from DNA microarray data sets. Biosystems 97: 44-59.
of websites by methods of bipartitionnement. Paris and ENSI, University of
Manouba.
and generic programming in the implementation of self organizing maps.
18. Benabdeslem K, Allab K (2013) Bi-clustering continuous data with self-
based topological maps. RNTI.