International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:3, No:1, 2009

BIDENS: Iterative Density Based Biclustering Algorithm With Application to Gene Expression Analysis

Mohamed A. Mahfouz, and M. A. Ismail

***Abstract*—**Biclustering is a very useful data mining technique for identifying patterns where different genes are co-related based on a subset of conditions in gene expression analysis. Association rules mining is an efficient approach to achieve biclustering as in BIMODULE algorithm but it is sensitive to the value given to its input parameters and the discretization procedure used in the preprocessing step, also when noise is present, classical association rules miners discover multiple small fragments of the true bicluster, but miss the true bicluster itself. This paper formally presents a generalized noise tolerant bicluster model, termed as μBicluster. An iterative algorithm termed as BIDENS based on the proposed model is introduced that can discover a set of k possibly overlapping biclusters simultaneously. Our model uses a more flexible method to partition the dimensions to preserve meaningful and significant biclusters. The proposed algorithm allows discovering biclusters that hard to be discovered by BIMODULE. Experimental study on yeast, human gene expression data and several artificial datasets shows that our algorithm offers substantial improvements over several previously proposed biclustering algorithms.

***Keywords*—**Machine learning, biclustering, bi-dimensional clustering, gene expression analysis, data mining.

I. Introduction

BICLUSTERING plays an important rule in analyzing the huge amount of valuable data produced by microarrays. Microarrays are molecular biology tools by which the expression patterns of thousands of genes can be monitored simultaneously. The gene expression data are organized as matrices where rows represent genes, columns represent various samples such as tissues or experimental conditions, and numbers in each cell characterize the expression level of the particular gene in the particular sample.

Given a data matrix D with set of rows R of size N and set of columns C of size M , biclustering can be defined formally as identifying a k possibly overlapping biclusters $B_l = (I_l, J_l)$ where I_l subset of R and J_l subset of C such that each bicluster B_l satisfies some specific characteristics of homogeneity.

Manuscript received September 15, 2008.

M. A. Mahfouz is a PhD candidate in the Department of Computer Science and Systems Engineering, Faculty of Engineering, Alexandria, 21544 Egypt (phone:+2012-481-2622; e-mail: mohamed.mahfouz @eng.alex.edu.eg).

M. .A. Ismail is Professor of Computer Science and Former Dean, Faculty of Enginneering, University of Alexandria, Alexandria 21455, Egypt.(He is now President of Pharos University in Alexandria) e-mail: m.a.ismail@pua.edu.eg.

Discovery of such clusters of genes is essential in revealing the significant connections in gene regulatory networks.

Many biclustering algorithms aimed at discovering biclusters with coherent values as in [1]-[4], [10], [11], [14], [17],[20]. Other biclustering algorithms address the problem of finding coherent evolutions across the rows and/or columns of the data matrix regardless of their exact values. According to their definition, a bicluster is a group of rows whose values induce a linear order across a subset of the columns, their work focuses on the relative order of the columns in the bicluster rather than on the uniformity of the actual values in the data matrix as in [5], [6], [9], [7], [12], [17].

Tanay *et al.,* 2002, [9], introduced SAMBA (Statistical-Algorithmic Method for Bicluster Analysis), a graph-theoretic approach to biclustering in combination with a statistical data model. In Samba framework, expression matrix is modeled as a bipartite graph, a bicluster is defined as a subgraph, and a likelihood score is used in order to assess the significance of observed subgraphs. It should however be noted that SAMBA's time complexity is $O(N\ 2^d)$ where d is the upper bound on the degree of each vertex.

The Coupled Two-Way Clustering (CTWC) introduced by Getz *et al.,* 2000, [1], and the Interrelated Two-Way Clustering (ITWC) introduced by Tang *et al.,* 2001, [2], follow straightforward way to identify biclusters by applying clustering algorithms to the rows and columns of the data matrix, separately, and then combining the results using some sort of iterative procedure to combine the two cluster arrangements.

Cheng and Church, 2000 [3], introduced similarity score called mean squared residue as a measure of the coherence of the rows and columns in the bicluster. The approach in [3] identify one bicluster at a time, mask it with random numbers, and repeat the procedure in order to eventually find other biclusters. Another approach Flexible Overlapped Biclustering (FLOC) introduced by Yang et al., 2003, [4]. It starts from a set of seeds (initial biclusters) and carries out an iterative process to improve the overall quality of the biclustering. At each iteration, each row and column is moved among biclusters to produce a better bi-clustering in terms of lower mean squared residues. After each iteration the best biclustering obtained will serve as the initial biclustering for the next iteration. The algorithm terminates when the current iteration fails to improve the overall biclustering quality.

Ihmels, *et al.,* 2004, [8], proposed a random Iterative

Signature Algorithm (ISA). Starting with an initial set of genes, all samples are scored with respect to this gene set and those samples are chosen for which the score exceeds a predefined threshold. In the same way, all genes are scored regarding the selected samples and a new set of genes is selected based on another user defined threshold. The entire procedure is repeated until the set of genes and the set of samples do not change anymore. The quality of the biclustering produced is highly dependent on the initial biclustering and the input parameters.

Prelic *et al.,* 2006, [10], proposed a divide-and-conquer algorithm (Bimax) for finding constant biclusters after discretizing the input expression matrix into a binary matrix. This discretization makes it harder to determine coherent biclusters.

Recently, Sharara and Ismail, 2007, [11], proposed an algorithm based on correlation termed as BISOFT. The algorithm identify one bicluster at a time by starting with initial one row , two columns bicluster and iteratively add a new row/column to the current bicluster such that this added row/column satisfy the criterion of having the average homogeneity within the bicluster above a pre-specified threshold for each dimension. The computational complexity is the main drawback of the algorithm. The cost of checking for adding a row is O ($N+M^2$) while for adding a column is O ($M+N^2$) .

The most relevant work to our focus here is the work done in [4], [14], [15], [20], [21]. Okada *et al.,* 2007, [20], proposed an exhaustive enumeration biclustering algorithm, (BIMODULE), based on closed itemset enumeration algorithm. The algorithm start by normalizing and discretizing the data matrix into L levels and the discretized data are given as input to closed itemset miner[13] in a form of transaction-items with the support Mg as a parameter.

A. B. Tchagang and A. H. Tewfik, 2005, [21], introduced novel biclustering algorithms using basic linear algebra and arithmetic tools, but the computational complexity is *O*(*NM LK*) where L is the number of distinct values in D

G. Liu *et al*., 2007, [14], introduced a distance-based subspace clustering model that uses a more flexible method to partition the dimensions to preserve meaningful and significant clusters may not be discovered by a grid based approach as in [20]. They proposed an algorithm based on closed itemset. Their algorithm considers only those biclusters containing a nontrivial number of objects and attributes, and do not mine for coherent bicluster.

In order to tackle those problems, a new iterative density based biclustering algorithm is proposed. The proposed algorithm iteratively add and remove rows and columns to initial biclusters similar to FLOC[4] but instead of minimizing the residue our objective was maximizing the density defined as the number of specified entries(not null) in a bicluster divided by its space defined as the multiplication of the average and the maximum range of bin numbers(after discretizing the input data matrix using histogram) used in each dimension. Our bicluster model extends the model in [14] to a generalized noise tolerant bicluster model using the concepts introduced by Pei *et al.*, [15], but unlike [14], our iterative procedure can mine coherent biclusters.

The paper is organized as follows: in section II the related works are illustrated; section III presents the proposed model and definitions; section IV describes the proposed algorithm, and section V reports on experimental results. Section VI is devoted to conclusions.

II. Related Works

A. FLOC Algorithm [4]

In [4] The residue of unspecified entry d_{ij} is zero while for specified entry is r_{ij} :

$$r_{ij} = d_{ij} - d_{iJ} - d_{Ij} + d_{IJ} \quad (1)$$

Where d_{iJ} is the average values of specified entries (not null) d_{ij} in row i for all columns $j \in$ J. d_{Ij} is the average values of specified entries d_{ij} in column j for all rows $i \in$ I. d_{IJ} is the average values of specified entries d_{ij} of the sub matrix defined by all rows $i \in$ I and all column $j \in$ J.
The residue of a bicluster(I,J) is r_{IJ} :

$$r_{IJ} = \sum_{i \notin I, j \in J} r_{ij}^2 / v_{IJ} \quad (2)$$

Where r_{ij} is the residue of an entry d_{ij} and v_{IJ} is the size of the bicluster defined as the number of specified(not null) entries d_{ij} such that $i \in$ I and $j \in$ J.

FLOC algorithm[4] starts from a set of seeds (initial biclusters) and carries out an iterative process to improve the overall quality of the biclustering. At each iteration, each row and column is moved among biclusters to produce a better biclustering in terms of lower mean squared residues. The best biclustering obtained during each iteration will serve as the initial biclustering for the next iteration. The algorithm terminates when the current iteration fails to improve the overall biclustering quality.

B. BIMODULE Algorithm [20]

The details of BIMODULE can be summarized as follow:-

1) Expression data from each micro array condition are linearly normalized to have mean 0 and variance 1, data farther than 3 standard deviation are regarded as outliers and are temporarily removed, and the rest of data are renormalized and if the normalized data contains new outliers the procedure is repeated until no outliers remain.
2) This normalized data is discretized such that the interval for each expression level is given by uniformly dividing the difference between the maximum and the minimum in the normalized data(without outliers) and outliers below the mean is assigned to the smallest level no. and outliers above the mean is assigned to the highest level no.
3) Each discretization level in each condition is given a unique number those numbers with the corresponding discretization level and condition form itemization table such that each row in the input expression data matrix will correspond to a transaction contains the corresponding items in that row excluding missed values. Transactions with their items are given as input to closed itemset miner[13] as input. Also two parameters Mg and Mc

correspond to minimum number of genes and minimum number of condition respectively in the output closed itemsets are given as input to the closed itemset miner[13].

4) The items in the enumerated closed itemset (the output of LCM) are converted to the condition name and discretized values by reference to the itemization table. Corresponding biclusters can be completed by selecting genes which match the required discretized values for each condition.
5) The enumerated biclusters are sorted using the following score F: $F(B)=A*\log_2(g)*\log_2(c)$ Where B is a bicluster and A represents the average of the absolute values of the discretized values in the conditions included g,c are the number of genes and conditions respectively. After sorting biclusters whose cells overlap by more than 25% with higher scoring bicluster are filtered out and the remaining biclusters are output to the user. Values chosen for L,Mg and Mc are 7,40 and 8 respectively .

III. Our Model of Bicluster

This section, formally presents a generalized noise tolerant bicluster model that can handle null values in a seamless manner(In the remaining of this paper, we use the term of biclusters to refer to the generalized biclusters).

This paper aims to discover four types of biclusters[17] as shown in fig (1).

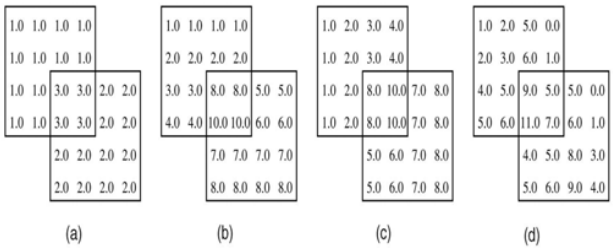

Fig. 1 Overlapping biclusters with general additive model. (a) Constant biclusters, (b) constant rows, (c) constant columns, and (d) coherent values

The data can be viewed as M×N matrix D of real numbers. Each entry d_{ij} in this matrix corresponds to the logarithm of the relative abundance of the mRNA of a gene i under a specific condition j, and may have a null value. The symbols *x, y,* · · · are used to denote a row in the data matrix D while *a, b,* · · · are used to denote a column in D.

Definition 1 (μ-BiCluster) : Let I a subset of rows and J a subset of columns in D. If for every two rows $x, y \in$ I and every attribute a $\in$ J, ($|v_{xa}-v_{ya}|<\mu$) or (v_{xa} is null) or (v_{ya} is null), then (I, J) form a μ-BiCluster.

Definition 2 (μ-CoherentBiCluster) : Let I a subset of rows and J a subset of columns in D. If for every two rows $x, y \in$ I and every two columns $a, b \in$ J, we have ($|(v_{xa}-v_{ya}) - (v_{xb}-v_{yb})|<\mu$) or ($v_{xa}$ is null) or (v_{ya} is null), then (I, J) form a μ-CoherentBiCluster.

It is clear that our model handles null values(unspecified) in a seamless manner such that missing values doesn't violate the constraints of accepted bicluster. Also in calculating the size of bicluster only the count of specified entries is considered.

Microarray data is subject to measurement noise, stemming from the underlying experimental technology and the stochastic nature of the studied biological behavior. In addition, uncertainty involved in choosing the proper thresholds when imputing discrete observations from the continuous gene expression values can introduce error.

To deal with noise, the definition of μ-BiCluster is modified as follow:

Definition 3 (μ-BiClusterWithNoise): Let I a subset of rows and J a subset of columns in D. If for every column $a \in$ J, there exist I'_a a subset of I such that for every two rows $x, y \in I'_a$, (($|v_{xa} - v_{ya}|<\mu$) or (v_{xa} is null) or (v_{ya} is null)) and $((|I| - |I'_a|)/|I| > \alpha)$ for every column a , and for every row x, the number of entries $|J'_x|$ not included in I's, $((|J| - |J'_x|)/|J| < \beta)$ then (I, J) form a μ-BiClusterWithNoise.

TABLE I
Example of Noisy Bicluster

	a	b	c	d
1	1	1	0	1
2	1	1	1	1
3	1	0	1	1
4	0	1	1	1
5	1	1	1	1
6	1	1	1	0

Table I illustrates how pattern in the data is obscured by noise. Classical frequent itemset algorithms will never produce the whole matrix as frequent term instead it will produce several smaller biclusters like({a, b, d),{1,2,5)} while using Definition 6 with α=β=0.25 the whole matrix is accepted μColumnwise-αβNoisyBiCluster bicluster since the percentage of ones in any row or column is greater than 0.75.

Definition 4 (space of μ-BiCluster) : The *space* of μ-BiCluster(I,J)is

$$\delta_{IJ} = ((\sum_{a\in J} \max_{x,y\in I} |v_{xa} - v_{ya}|)/|J|) \max_{a\in J} \max_{x,y\in I} |v_{xa} - v_{ya}| \quad (3)$$

Definition 5 (size of Bicluster) : the size of a bicluster v_{IJ} is defined as the number of specified entries d_{ij} such that i $\in$ I and j $\in$ J.

IV. The Proposed Algorithm

Finding an exact solution for the biclustering problem could be time consuming because the problem is NP-hard as proven in [3]. In this section, we present an iterative algorithm termed as BIDENS, which can efficiently and accurately approximate a k possibly overlapping biclusters. The algorithm start with k initial biclusters (accepted *u*biclusters) and iteratively move rows and columns from or to biclusters such that the resulting biclustering is accepted and the average space of biclusters

defined in section III, is minimized from one iteration to the next otherwise it terminates. The model is based on the real values of the input matrix, to reduce the complexity of the proposed algorithm we will discretize the matrix such that the threshold μ will be an integer value corresponds to the range of bins the values of each columns lie in similar to BIMODULE[20], but unlike[20], the discretization will be done to the whole values in the input matrix using histogram and a level will correspond to several contagious bins(possibly overlapping with another level).

A. Discretization

This section describes the discretization scheme which is the preprocessing step for our algorithm. One has to note that this step is optional; we can work directly on the values of entries in D as in the definitions and the threshold μ will be a real number depends on the range of values in each column. In our experiments we used a grid based approach to partition the data space into small rectangle cells as in the density based subspace clustering algorithms. The procedure starts by computing histogram for all values in the data matrix D. Instead of using fixed level as in the discretization procedure of BIMODULE, the level in our approach corresponds to a set of contiguous bins (possibly overlaps with another level) in the histogram representing the data in D. the threshold μ given as input parameter to the algorithm will be an integer.

It is desirable to compute a histogram that provide good resolution but also have data artifacts smoothed out. A number of studies have addressed the problem of how many equi-width bins can be supported by a given distribution [16].

Based on these studies, a reasonable, simple approach would be to make the number of equi-width bins inversely proportional to the standard deviation of the data values and directly proportional to $N^{1/3}$, where N is the number of points to compute a histogram for. Alternatively, one can use a global binning strategy and coarsen a histogram as the number of points decreases. In BIDENS global binning strategy was chosen. BIDENS is robust with respect to different binning strategies as long as the histograms do not significantly undersmooth or oversmooth the distribution density. Expression data matrices is large datasets and have the advantage of supporting the computation of detailed histograms with good resolution**.**

After a histogram is computed its bins are numbered contiguously and bins have count less than 5% of the global uniform level are identified as outlier bins then outlier bins reassigned numbers equal to the number of the nearest not outlier bin. Each level represented by a set of contiguous bins.

For example suppose 2,3,110,60,40,1,2,70,120,20,3,1 were the counts of bins in a histogram has 12 bins. Since (408/12=34) is the global uniform level then we have 6 outlier bins. Then the histogram bins will be numbered 3,3,3,4,5,5,8,8,9,10,10,10

If we choose 3 levels then (12-6)/3=2 is the maximum difference between two bins to lie in the same level. Data values that lies in bin number 1-4, 4-6, 7-9, 9-12 will be considered in the same level. As we can see our discretization procedure can deal with outlier in the middle of the distribution like 3 and 10, Whereas, the discretization procedure used in [20] deals only with outliers at the boarder of the distribution.

B. Description of the Proposed Algorithm

The objective of the proposed algorithm is to maximize the average size and minimize the average space of the produced biclusters.

Both the average and the maximum is included in the definition of the space in previous section to allow measuring the effect of an action that may cause high increasing in the maximum while the average slightly change in large bicluster.

Suppose we have μ-CoherentBiCluster bicluster and we need to add a new row to it and check the result as accepted μ-CoherentBiCluster. It is clear from definition 2 that the computational complexity is O (2^m n) where m and n is the number of rows and columns of the bicluster respectively.

To reduce the computational complexity we will use the approach used in [21]. A bicluster B with coherent values can be viewed as the sum of three matrices: B_1 with constant values, B_2 with constant values on rows, and B_3 with constant values on columns, that is, $B = [\mu+\alpha_i+\beta_j] = [\mu] + [\alpha_i] + [\beta_j]$, with $B_1 = [\mu]$, $B_2 = [\alpha i]$ and $B_3 = [\beta_j]$. Therefore, to obtain perfect biclusters with coherent values, the following approach can be used. To check for a μCoherentBicluster(I,J) we can simply check for a μBicluster after subtracting constant row bicluster from it. The constant row matrix can be constructed using in column j∈J.

Since we do not have any knowledge about the rows of the gene expression matrix D, the intuitive approach is to use an iterative multistep approach but to simplify the computation we try a few columns and found this enough for perfect coherent bicluster as the following example will show.

TABLE II
EXAMPLE OF μCOHERENTBICLUSTER

401	120	298
318	37	215
322	41	219

a) original data

9	3	6
7	1	4
8	2	5

b) discretized data

6	0	3
6	0	3
6	0	3

c)subtracted from column #2

The matrix in Table II (a) is disritized in Fig. II(b) such that each number correspond to bin number.

If we choose u=3 then the matrix in Fig. II(b) represents μBicluster such that : The average number of bins used in all columns is ((9-7+1) + (3-1+1) + (6-4+1))/3 =3, maximum number of bins used in every column=3, the space of the bicluster = 3*3=9, the size of the bicluster =9

However if we choose u =1 the same matrix is not μBicluster but it is clear that if we subtract column #2 from each column in the matrix the resulting matrix is μCoherentBicluster with u=1.

As the above example show if we construct the constant row matrix which to be subtracted from the original matrix using any column of the above matrix in table II(b) the result

is is μBicluster with u=1 as in table II(c). we do not need to define μ-CoherentBiclusterWithNoise since it is enough to test the matrix resulting from subtraction as μ-BiClusterWithNoise.

TABLE III
ALGORITHM BIDENS

Begin [BIDENS]
1. Discretize the matrix as in section IV(A)
2. Generate k initial acceptable μBicluster save as best biclustering.
3. d=1
4. **Repeat**
   4.1. **if** d=1 **Set** μBicluster as the current definition of bicluster
   4.2. **if** d=2 **Set** μCoherentBicluster as the current definition of bicluster.
   4.3. **if** d=3 **Set** μCoherentBiclusterWithNoise as the current definition of bicluster.
   4.4. Determine the best action for each row and each column using (4) and using the current definition of bicluster.
   4.5. Partially order the actions by swapping
   4.6. Perform the best action for every row and column sequentially
   4.7. Select the minimum average space from accepted biclustering obtained in 6 that has a larger size than best biclustering as best clustering, Otherwise there is no improvement set d=d+1.
   Until (d>3)
5. Output best biclustering and terminate.

End [BIDENS]

After the initialization phase an iterative process is started to improve the quality of the biclusters continuously. During each iteration in the second phase, each row and each column are examined to determine its best action towards reducing the overall average space of biclusters. These actions are then performed successively to improve the biclustering. An action is defined with respect to a row (or column) and a bicluster. There are *k* actions associated with each row (or column), one for each bicluster. For a given row (or column) *x* and a bicluster *c*, the action Action(*x,c*) is defined as the change of membership of *x* with respect to *c*. Note that this action is uniquely defined at any stage. If *x* is already included in *c* then Action(*x,c*) represents the removal from the bicluster c. Otherwise, Action(x,c) denotes the addition of x to the bicluster c.

The gain of an action is defined as a function of the relative reduction in c's space and the relative enlargement of c's size as a consequence of performing the action.

Definition 6 Gain of an action, Gain(x,c) :
Given a space threshold δ the gain of an action Action(x,c) is defined as $\mathrm{Gain}(x,c) = \delta_{old}(\delta_{old} - \delta_{new})/\delta^2 + (v_{new} - v_{old})/v_{old}$ (4)

δ_{new}, δ_{old} are the spaces of a bicluster c before and after executing the action c. Similarly v_{old}, v_{new} are the sizes of the bicluster c before and after executing the action.
If we define a level as μ contagious bins then $\delta = \mu^2$.
In the example given in Table 3, if we are mining for μBicluster(d=1), suppose that we need to compute the gain of adding row #2 to the bicluster({1,3},{1,2,3}) using the definition in (4) and that μ=3 is as follow*:*
The gain=4 (4-9)/81+(9-6)/6= 0.254
While if we are mining for μCoherentBicluster(d=2), the gain of adding row #2 to the bicluster({1,3},{1,2,3}) using the definition in (4) :
The gain=1 (1-1)/81+(9-6)/6= 0.5

The gain of the same action may change from one phase to another depending on the definition of accepted bicluster.

After the best action is identified for every row (or column), these actions are then performed sequentially. The best biclustering obtained during the last iteration, denoted by *best_biclustering* is used as the initial biclustering of the current iteration.

After mining for μBiclusters then we start another phase in which we mine for μCoherentBicluster after that in the last phase we accept μCoherentBiclustersWithNoise.

Finally step 4.7 may be modified to output the current biclustering before incrementing d.

C. Complexity Analysis

Recall that N is the number of rows of the gene expression matrix D, *M* is the number of columns in D and H is the number of histogram bins used in discretizing D where H=L*B where L is the discretization levels and B is the number of representative bins for each level while k is the number of biclusters to be found.

Discretization phase for BIDENS requires (NM+(N+M)*B).

Memory requirement for both BIDENS and FLOC is proportional to (N+M) for keeping the lower and upper limits for rows and columns of biclusters in BIDENS and the averages of columns and rows for FLOC.

The computational complexity of BIDENS is similar to FLOC i.e. O $(N+M)^2$ K p as reported in [4], where p is the number of iterations but the computation of the gain in our algorithm includes simple comparisons while in [4] includes addition, subtraction and squaring the computed residues. Note that the two algorithms has less computational complexity than CC algorithm as it is typically the case where (N+M) >> p.

The computation time for finding frequent itemsets depends on the number of itemsets found [13], and the minimum support Mg. When the minimum support is large and the number of the itemsets found is small, the computation time for each itemset is relatively large. However ,computation time per itemset decreases as the increase of the itemsets found, and roughly speaking when the size of output file is equal to the input database size, it will be constant. The memory usage of LCM is always linear in the size of the input database. Approximately LCM uses integers at most three times as much as the database size, which is the sum of the number of items of each transaction.

V. Experimental Results

A. Synthetic Overleaped datasets

In this experiment the input to the biclustering algorithm is the artificial dataset[19] used in [20] to test the ability of BIMODULE in discovering highly overlapped biclusters.

Each matrix in the dataset is a binary matrix with 10 modules(biclusters) are implanted into a background matrix, they consider 11 different overlapping degree(d=0,1,2..10) where the size of the background matrix and modules vary from 100×100 to 110×110 and from 10×10 to 20×20, respectively.

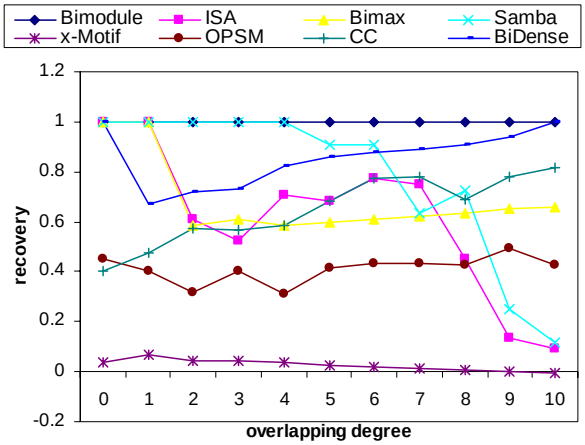

Fig 2 The recovery for overlapping biclusters

Figs. 3-4 show the recovery and relevance respectively of several algorithms. Even though BIDENS is initialized by biclusters produced by BIMODULE we found that BIMODULE outperforms other algorithms including BIDENS.

The higher recovery reported by BIMODULE is expected since BIMODULE for two largely overlapped biclusters will most probably produce three bicluster corresponds to the two true biclusters and a third one corresponds to the intersection of the two overlapped biclusters BIDENS which make its recovery higher than other algorithms for overlap degree higher than 8 and relevance slightly the same for different overlap degree.

In iterative approach like CC or BIDENS the probability of missing a row or a column in the true biclusters is almost the same as adding a row or column not in the true biclusters this explains why the relevance close to the recovery. Even though the binary datasets used in the experiments does not give chance to our algorithm to use the flexibility of our bicluster model but the results show that our iterative approach can deal with overlapping.

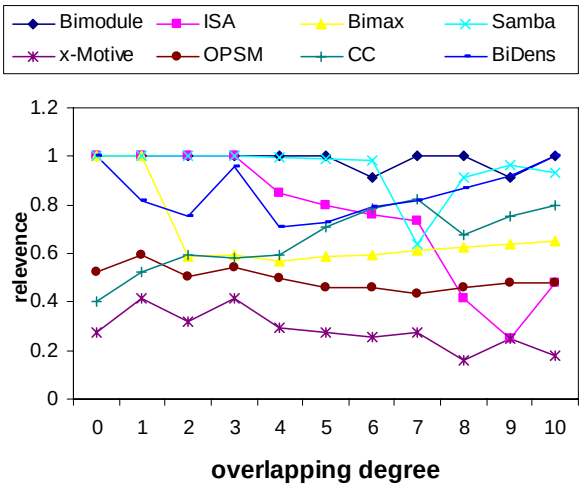

Fig. 3 The relevance for overlapped biclusters

B. Yeast Dataset

In this experiment we discretize the whole matrix representing the Yeast dataset in [18] using histogram such that each level correspond to 10 bins in the histogram (μ=10) such that each column's values lie in number of levels equal or greater than threshold λ.

TABLE IV
Performance Comparison on Yeast Dataset

Algorithm	avg. residue	avg. size	avg. gene.	avg. cond.
CC	204.293	1576.98	167	12
FLOC	187.543	1825.78	195	12.8
BIMODULE	330.65	1036.86	240.97	5.08
BIDENS	191.38	2517.42	248	13.1

The results reported in Table IV for λ =9. The higher the value of λ the smaller the biclustering produced and the higher the quality.

As shown in Table IV the biclusters produced by BIDENS is larger and has less average residue than biclusters of any of the other three algorithms.

It is clear that a matrix satisfies definition 1-3 is a coherent bicluster with an error added to its entries depend on the range of values each level represent. As the range that a level represents decreases the residue expected to decreases and would reach zero if the range was zero.

The 100 biclusters reported for BIMODULE 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 BIMODULE we need to use discretization levels greater than 7 which is the number of levels recommended in [20] which will result in longer runtime and smaller bicluster size.

C. Human Dataset

Table V shows summary of the results obtained in experiments with Human dataset. The results are similar to the results obtained with Yeast dataset. BIDENS produces larger biclusters with lower average results.

TABLE V
PERFORMANCE COMPARISON ON HUMAN DATASET

Algorithm	avg. residue	avg. size	avg. gene	avg. cond.
CC	850.04	4456.43	269.22	24.5
FLOC	795	5859.68	276.4	26.5
BIMODULE	1022	3250.44	297.78	13.8
BIDENS	759.83	6361.45	284.75	32.18

BIMODULE give the worst results with this large dataset. Results of BIDENS is almost close to FLOC algorithm.

VI. CONCLUSION

In this paper a new iterative density based algorithm is proposed which simplify the computation used in the iterations of FLOC algorithms by computing the density instead of residue. Dense sub matrices are expected to have low residue as the experiments show. From experimental results and algorithm analysis, the following points can be concluded:

1) Increasing the number of levels in BIDENS may increase the quality of the output clusters while reduces the average size as expected, while, in BIMODULE increasing the number of levels may reduce the quality since it reduces the total number of biclusters produced by the algorithm.
2) Results of BIMODULE for real dataset show that the quality of its output in terms of low average residues is lower than those obtained for BIDENS, FLOC or CC.
3) BIMODULE give better results for overlapped datasets for all overlapping value.
4) BIMODULE is faster than BIDENS, FLOC or CC.
5) The runtime of BIDENS is higher than BIMODULE but compares favorably with FLOC and CC.
6) Iterative approach allows for additional features similar to FLOC, like forcing constraints on the output biclusters.
7) The number of biclusters produced by closed itemset miner is huge and hard to select best biclustering from them.
8) In BIMODULE, The runtime increases as the values of Mg and Mc increase while in BIDENS they do not effect the runtime.
9) In our algorithm small and large biclusters can be discovered because no constraints like min gene (mg) as in closed itemset.
10) BIMODULE can be used to initialize other iterative techniques like FLOC or our algorithm also it can help in estimating suitable values for the input parameters of other algorithms.

REFERENCES

[1] G. Getz, E. Levine, and E. Domany, "Coupled Two-Way Clustering Analysis of Gene Microarray Data," Proc. Natural Academy of Sciences US, pp. 12079–12084, 2000.
[2] C.Tang, L.Zhang, I.Zhang, and M.Ramanathan, "Interrelated Two-Way Clustering: An Unsupervised Approach for Gene Expression Data Analysis," Proc. Second IEEE Int'l Symp. Bioinformatics and Bioeng., pp. 41–48, 2001.
[3] Y. Cheng and G. Church, "Biclustering of expression data," Proc. Eighth Int'l Conf. Intelligent Systems for Molecular Biology(ISMB '00), pp. 93–103, 2000.
[4] J. Yang, W. Wang, H. Wang, and P. Yu, "Enhanced Biclustering on Expression Data," Proc. Third IEEE Conf. Bioinformatics and Bioeng.,pp. 321–327, 2003.
[5] T.M. Murali and S. Kasif, "Extracting Conserved Gene Expression Motifs from Gene Expression Data," Proc. Pacific Symp. Biocomputing,vol. 8, pp. 77–88, 2003.
[6] L. Lazzeroni and A. Owen, "Plaid Models for Gene Expression Data," technical report, Stanford Univ., 2000.
[7] A. Ben-Dor, B. Chor, R. Karp, and Z. Yakhini, "Discovering Local Structure in Gene Expression Data: The Order-Preserving Submatrix Problem," Proc. Sixth Int'l Conf. Computational Biology (RECOMB '02), pp. 49–57, 2002.
[8] J. Ihmels, S. Bergmann, and N. Brkai, "Defining Transaction Modules using large scale gene expression data," Bioinformatics,Vol.20,No.13,pp.1993–2003, 2004.
[9] A. Tanay, R. Sharan, and R. Shamir, "Discovering Statistically Significant Biclusters in Gene Expression Data," Bioinformatics, vol. 18, pp. S136–S144, 2002.
[10] A. Prelic, S. Bleuler, P. Zimmermann, A.Wille, P. Buhlmann, W. Gruissem, L. Hennig, L. Thiele, and E.Zitzler, "A Systematic comparison and evaluation of biclustering methods for gene expression data," Bioinformatics, 22:1122–1129, 2006.
[11] H. Sharara M.A.Ismail, "αCORR: A novel algorithm for clustering gene expression data," Bioinformatics and Bioengineering, 2007. BIBE 2007. Proceedings of the 7th IEEE International Conference, pp. 974–981, 2007.
[12] J. Liu and W. Wang, "OP-Cluster: Clustering by Tendency in High Dimensional Space," Proc. Third IEEE Int'l Conf. Data Mining, pp. 187-194, 2003.
[13] LCM ver2 Available http://research.nii.ac.jp/~uno/codes-j.html.
[14] G. Liu,Jinyan, L. Kelvin and L. Wong, "Distance Based Subspace Clustering with Flexible Dimension Partitioning," IEEE, pp. 1250–1254, 2007.
[15] J. Pei, A. K. Tung, and J. Han., "Fault-tolerant frequent pattern mining: Problems and challenges,"Workshop on Research Issues in Data Mining and Knowledge Discovery, 2001.
[16] M. P. Wand, "Data-Based Choice of Histogram Bin Width," The American Statistician, vol. 51, 1996, pp. 59–64.
[17] Sara C. Madeira and Arlindo L. Oliveira, "Biclustering Algorithms for Biological Data Analysis: A Survey," IEEE TRANS. Computational Biology And Bioinformatics, vol. 1, 2004.
[18] Yeast and Human Dataset. Available http://arep.med.harvard.edu/network discovery.
[19] SyntheticDatasets. Available http://www.tik.ee.ethz.ch/sop/bimax/SupplementMatrials,Biclustering.html.
[20] Y. Okada, W. Fujibuchi and P. Horton, "Module Discovery in Gene Expression Data Using Closed Itemset Mining Algorithm," IPSG transactions in bioinformatics, vol.48, pp39–48, 2007.
[21] A. B. Tchagang and A. H. Tewfik, "DNAMicroarray Data Analysis: A Novel Biclustering Algorithm Approach," EURASIP Journal on Applied Signal Processing, vol. 2006, pp. 1–12.