BIOINFORMATICS ORIGINAL PAPER

Vol. 23 no. 1 2007, pages 50–56
doi:10.1093/bioinformatics/btl560

Gene expression

Computing the maximum similarity bi-clusters of gene expression data

Xiaowen Liu and Lusheng Wang*

Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong

Received on September 2, 2006; revised and accepted October 31, 2006
Advance Access publication November 7, 2006
Associate Editor: Martin Bishop

ABSTRACT

Motivations: Bi-clustering is an important approach in microarray data analysis. The underlying bases for using bi-clustering in the analysis of gene expression data are (1) similar genes may exhibit similar behaviors only under a subset of conditions, not all conditions, (2) genes may participate in more than one function, resulting in one regulation pattern in one context and a different pattern in another. Using bi-clustering algorithms, one can obtain sets of genes that are co-regulated under subsets of conditions.

Results: We develop a polynomial time algorithm to find an optimal bi-cluster with the maximum similarity score. To our knowledge, this is the first formulation for bi-cluster problems that admits a polynomial time algorithm for optimal solutions. The algorithm works for a special case, where the bi-clusters are approximately squares. We then extend the algorithm to handle various kinds of other cases. Experiments on simulation data and real data show that the new algorithms outperform most of the existing methods in many cases. Our new algorithms have the following advantages: (1) no discretization procedure is required, (2) performs well for overlapping bi-clusters and (3) works well for additive bi-clusters.

Availability: The software is available at http://www.cs.cityu.edu.hk/~liuxw/msbe/help.html.

Contact: lwang@cs.cityu.edu.hk

Supplementary information: The Supplementary Data is available at http://www.cs.cityu.edu.hk/~liuxw/msbe/supp.html.

1 INTRODUCTION

The advent of microarray technologies has made the experimental study of gene expression faster and more efficient. Microarrays have been used to study different kinds of biological processes. The microarray experiments are carried on a genome with a number of different conditions (samples), such as different time points, different cells or different environmental conditions (Baldi and Hatfield, 2002). The data from microarray experiments is usually in the form of large matrices, in which each row corresponds to a gene, each column corresponds to a condition, and each entry denotes an expression level of a gene under a condition.

A lot of analysis techniques have been proposed for identifying a subset of genes sharing compatible expression patterns. Different from traditional clustering methods, such as hierarchical clustering and *k*-means clustering, Cheng and Church (2000) used a bi-clustering method for the analysis of gene expression data. Bi-clustering has shown its usefulness and advantages in many applications. The underlying bases for using bi-clustering in the analysis of gene expression data are (1) similar genes may exhibit similar behaviors only under a subset of conditions, not all conditions, (2) genes may participate in more than one function, resulting in one regulation pattern in one context and a different pattern in another.

Many bi-clustering methods have been proposed in recent years. Madeira and Oliveira (2004) discussed several types of bi-clusters. Prelić *et al*. (2006) gave a systematic comparison of different bi-clustering methods. Tanay *et al*. (2002) and Prelić *et al*. (2006) focused on finding bi-clusters of up-regulated expression values or down-regulated expression values. They discretized the original expression matrices and their bi-clustering methods work on binary matrices. In the gene expression analysis, people are more interested in finding a subset of genes showing similar up and down regulations under a subset of conditions. Ihmels *et al*. (2002, 2004) used gene signature and condition signature to find bi-clusters with both up-regulated and down-regulated expression values. When no a priori information of the matrix is available, they proposed a random iterative signature algorithm (ISA). Cheng and Church (2000) defined a mean squared residue function to measure the quality of a bi-cluster. They also gave the concept of δ-biclusters and a greedy algorithm for finding δ-biclusters. Yang *et al*. (2002) improved Cheng and Church's method by allowing missing gene expression values in gene expression matrices. Ben-Dor *et al*. (2002) proposed to find the order-preserving sub-matrix (OPSM) in which all genes have same linear ordering and gave a heuristic algorithm for the OPSM problem. Murali and Kasif (2003) presented a random algorithm, XMOTIF.

In this paper, we define a similarity score between two genes and define a similarity score for a sub-matrix. We believe that this is the first time that similarity score is used for solving bi-clustering problem. Using the similarity score, we design a polynomial time algorithm to find an optimal bi-cluster. To our knowledge, this is the first formulation for bi-cluster problems that admits a polynomial time algorithm for optimal solutions. The algorithm works for a special case, where the bi-clusters are approximately squares. We then extend the algorithm to handle various kinds of other cases. Experiments on simulation data and real data show that the new algorithms outperform most of the existing methods in many cases. Our new algorithms have the following advantages: (1) no discretization procedure is required, (2) performs well for overlapping bi-clusters and (3) works well for additive bi-clusters.

*To whom correspondence should be addressed.

1.0	2.0	3.0	4.0
1.0	2.0	3.0	4.0
1.0	2.0	3.0	4.0
1.0	2.0	3.0	4.0

(a)

1.0	2.0	5.0	0.0
2.0	3.0	6.0	1.0
4.0	5.0	8.0	3.0
5.0	6.0	9.0	4.0

(b)

Fig. 1. Examples of two types of bi-clusters. The bold rows are the reference genes. (**a**) Constant bi-cluster. (**b**) Additive bi-cluster.

2 METHODS AND ALGORITHMS

Let $A(I,J)$ be an $n \times m$ matrix of real numbers, where $I = \{1, 2, \ldots, n\}$ is the set of genes and $J = \{1, 2, \ldots, m\}$ is the set of conditions. The element a_{ij} of $A(I,J)$ represents the expression level of gene i under condition j. For gene subset $I' \subseteq I$ and condition subset $J' \subseteq J$, $A(I',J')$ denotes the sub-matrix (bi-cluster) of $A(I,J)$ that contains only the elements a_{ij} satisfying $i \in I'$ and $j \in J'$.

In some cases, the reference gene we are interested in is known in advance. Our goal is to find a subset of genes that are related to the reference gene. When the reference gene is not known, we can enumerate all genes in the matrix or randomly select a number of genes as the reference genes. Similar ideas are also used in Ihmels *et al*. (2004).

CONSTANT BI-CLUSTERS AND ADDITIVE BI-CLUSTERS. *Let $A(I,J)$ be an $n \times m$ gene expression matrix and $i^* \in I$ a reference gene. A bi-cluster $A(I', J')$ with $I' \subseteq I$ and $J' \subseteq J$ is a constant bi-cluster for reference gene i^* if for any $i \in I'$ and any $j \in J'$, $a_{ij} = a_{i^*j}$. A sub-matrix $A(I',J')$ with set of rows I' and set of columns J' is an additive bi-cluster for reference gene i^* if for any $i \in I'$ and any $j \in J'$, $a_{ij} - a_{i^*j} = c_i$, where c_i is a constant for any row i.*

Figure 1a gives an example of a constant bi-cluster, where every row in the sub-matrix is identical. Figure 1b gives an example of additive bi-cluster.

First, we define a similarity score to measure the similarity between the reference gene and any other genes.

2.1 Similarity score between genes

For an element a_{ij} of expression matrix $A(I,J)$ and a reference gene $i^* \in I$, define $d_{ij} = |a_{ij} - a_{i^*j}|$. When finding constant bi-clusters, we want to ignore elements with big d_{ij}. So we set a threshold $\alpha \cdot d_{\text{avg}}$, where

$$d_{\text{avg}} = \frac{\sum_{i \in I \& j \in J} d_{ij}}{|I||J|}$$

is the average distance value of all elements in $A(I,J)$. If $d_{ij} \geq \alpha \cdot d_{\text{avg}}$, we believe that the two elements a_{ij} and a_{i^*j} are not similar and set the similarity s_{ij} to be 0. Otherwise, the similarity score is

$$1 - \frac{d_{ij}}{\alpha \cdot d_{\text{avg}}} + \beta$$

where β is the bonus for small d_{ij}. The purpose for using β is to further enlarge the similarity score for small d_{ij} and ignore d_{ij}'s that are greater than the threshold. That is, we define

$$s_{ij} = \begin{cases} 0 & \text{if } d_{ij} > \alpha \cdot d_{\text{avg}} \\ 1 - \dfrac{d_{ij}}{\alpha \cdot d_{\text{avg}}} + \beta & \text{otherwise.} \end{cases} \quad (1)$$

When $d_{ij} \leq \alpha \cdot d_{\text{avg}}$, we have $\frac{d_{ij}}{\alpha \cdot d_{\text{avg}}} \leq 1$. Thus, s_{ij} is always greater than or equal to 0.

We use $S(I,J)$ to denote the $n \times m$ similarity matrix containing the set of rows I and the set of columns J with every element s_{ij} computed as in Equation (1).

2.2 Similarity score for a bi-cluster

Let $S(I,J)$ be an $n \times m$ similarity matrix and $S(I',J')$ be a bi-cluster (sub-matrix) of $S(I,J)$. For row $i \in I'$, the similarity score of row i in $S(I', J')$ is $s(i,J') = \sum_{j \in J'} s_{ij}$. For column $j \in J'$, the similarity score of column j in $S(I',J')$ is $s(I',j) = \sum_{i \in I'} s_{ij}$. The similarity score of $s(I',J')$ is $s(I',J') = \min\{\min_{i \in I'} s(i,J'), \min_{j \in J'} s(I',j)\}$.

Algorithm 1 The MSB Algorithm

Input An $n \times m$ similarity matrix $S(I,J)$.
Output A maximum similarity bi-cluster $S(I_A, J_A)$.

1. Set the first bi-cluster $S(I_1, J_1) = S(I,J)$ and compute the similarity score for all rows and columns of $S(I_1, J_1)$.
2. **For** $k = 1$ to $n + m - 2$ **do**
3. Find row $i' \in I_k$ such that $s(i', J_k) = \min_{i \in I_k} s(i, J_k)$.
4. Find column $j' \in J_k$ such that $s(I_k, j') = \min_{j \in J_k} s(I_k, j)$.
5. **If** $s(i', J_k) < s(I_k, j')$, **then** set $I_{k+1} = I_k - \{i'\}$ and $J_{k+1} = J_k$, **else** set $I_{k+1} = I_k$ and $J_{k+1} = J_k - \{j'\}$
6. Let $S(I_{k'}, J_{k'})$, $1 \leq k' \leq n+m-1$, be the bi-cluster such that $s(I_{k'}, J_{k'}) = \max_{1 \leq k \leq n+m-1} s(I_k, J_k)$.
7. Output $S(I_A, J_A) = S(I_{k'}, J_{k'})$.

Fig. 2. The MSB algorithm.

Consider a constant bi-cluster $S(I',J')$. If the similarity score of row $i \in I'$ in $S(I',J')$ is high, gene i has similar expression values with the reference gene i^* under the column subset J'. If the similarity score of column $j \in J'$ in $S(I',J')$ is high, the expression values in column j of all genes in I' are similar to that of the reference gene i^*. Thus, to find a constant bi-cluster, we want to find a sub-matrix $S(I',J')$ with the highest similarity score $s(I',J')$.

DEFINITION 1. *Given an $n \times m$ similarity matrix $S(I,J)$, the maximum similarity bi-cluster problem (MSB) is to find a bi-cluster $S(I',J')$ with $I' \subseteq I$ and $J' \subseteq J$ such that $s(I',J')$ is maximized. The bi-cluster $S(I',J')$ is called the maximum similarity bi-cluster of $S(I,J)$.*

From the definition $s(I',J') = \min\{\min_{i \in I'} s(i,J'), \min_{j \in J'} s(I',j)\}$, it seems that the sub-matrices $S(I', J')$ that are approximately squares will have big $s(I',J')$ value. Consider a sub-matrix $S(I',J')$ with $|I'| \gg |J'|$ [the number of rows is much bigger than the number of columns in $S(I',J')$]. In this case, the value of $\min_{j \in J'} s(I',j)$ should be much larger than the value of $\min_{i \in I'} s(i,J')$, since the number of numbers in a column in J' is much bigger than the number of numbers in a row in I'. Therefore, in this case, $s(I',J') = \min_{i \in I'} s(i,J')$ and the columns with higher score do not help. To get a sub-matrix with better score, we can delete some rows from I' with smallest scores. Suppose $I'' \subset I'$ is obtained from I' by deleting a few rows. Then $\min_{i \in I''} s(i,J') \geq \min_{i \in I'} s(i,J')$. Thus, if $|I'| \gg |J'|$, we can obtain a sub-matrix with better similarity score by deleting some rows in I'. Similarly, if $|J'| \gg |I'|$, we can get a a sub-matrix with better similarity score by deleting some columns in J'. This shows that $|I'|$ and $|J'|$ should not be quite different. Our simulation results also show this point.

2.3 The exact algorithm

In this section, we present a polynomial time algorithm for finding the maximum similarity bi-cluster in an $n \times m$ similarity matrix $S(I,J)$. The algorithm is in fact a greedy algorithm. The sketch is as follows: (1) We start with the whole matrix as the bi-cluster. (2) We then delete the row or the column whose similarity score is the smallest (the worst) among all rows and columns in the current bi-cluster. (3) We repeat the above process until there is one element in the current bi-cluster. (4) During this process, we obtain $n + m - 1$ bi-clusters. Among the $n + m - 1$ sub-matrices, we choose the sub-matrix $S(I_{k'}, J_{k'})$ that has the maximum similarity score $s(I_{k'}, J_{k'})$. The algorithm is named as the MSB algorithm and is shown in Figure 2.

In Step 1, the time complexity for computing the similarity scores of all rows and columns of $S(I_1, J_1)$ is $O(n \times m)$. Step 3–5 is repeated $O(n + m)$ times. Steps 3 and 4 can be done in $O(n + m)$ time if we use $O(1)$ time to compute the similarity score of a row or a column in a new matrix by deleting

one row or one column. In fact, $O(1)$ time is enough to get the similarity score of a row or a column in the new matrix (by deleting a row or a column) if we use the similarity score of the old matrix (before deleting the row or the column). Obviously, Step 5 can be done in $O(1)$ time. Therefore, the time complexity of the whole algorithm is $O((n+m)^2)$.

THEOREM 1. *The MSB algorithm runs in $O((n+m)^2)$ time and outputs an optimal solution for the MSB problem.*

The proof of the theorem is given in the Supplementary material.

The MSB algorithm can find the optimal solution for the MSB problem. But in some cases, the bi-clusters discovered by the MSB algorithm are large and they contain lots of elements with low similarity values. Thus, we also want to find bi-clusters in which most of the elements have high similarity scores. We introduce the average similarity score as the second criterion for controlling the qualities of bi-clusters. Given a bi-cluster $S(I',J')$ of $S(I,J)$, the average similarity score of $S(I',J')$ is

$$s_{\text{avg}}(I',J') = \frac{\sum_{i\in I'} \sum_{j\in J'} s_{ij}}{|I'||J'|}.$$

Let γ be the threshold of the average similarity score. We want to find the bi-cluster whose average similarity score is no less than γ and the similarity score of the bi-cluster is maximized. In this case, the threshold γ is an input parameter. We slightly modify the MSB algorithm to handle the threshold γ: only the bi-clusters with average similarity scores no less than γ are considered in step 6 of the MSB algorithm. That is, step 6 is modified as follows: let $S(I_{k'}, J_{k'})$, $1 \le k' \le n+m-1$, be the bi-cluster such that

$$s(I_{k'},J_{k'}) = \max_{1\le k\le n+m-1 \;\&\; s_{\text{avg}}(I_k,J_k)\ge\gamma} s(I_k,J_k).$$

The modified algorithm is named γ-MSB algorithm.

2.4 Extension algorithm

The γ-MSB algorithm tends to find bi-clusters that are approximately squares. When the number of rows and number of columns in the bi-cluster are quite different, the γ-MSB algorithm can only find a sub-matrix with some rows or some columns missing. In practice, there are thousands of genes (rows) in the given matrix. So, in most cases, some rows are missing. To solve the problem, we do the following: (1) Use the γ-MSB algorithm to find a bi-cluster $S(I',J')$ in $S(I,J)$. (2) With the subset of conditions $J' \subseteq J$, we extend the bi-cluster by adding rows into the bi-cluster. If a row $i \in I$ has high similarity scores in the subset of condition J', i.e. $s(i,J') \ge \gamma_e \cdot |J'|$, we add row i into the bi-cluster. Here $\gamma_e \cdot |J'|$ is a threshold to control the quality of the final bi-cluster.

When the reference gene is not known, we can enumerate every row in I as the reference gene. We refer to this algorithm as constant MSBE algorithm.

2.5 Additive bi-clusters

Let $A(I,J)$ be the input matrix. In an additive bi-cluster $A(I',J')$ with $I' \subseteq I$ and $J' \subseteq J$, the expression values of the genes fluctuate in the same way as the reference gene. In an error-free additive bi-cluster $A(I',J')$ for reference gene i^*, we have $\forall i \in I'$ and $\forall j \in J'$, $a_{ij} = a_{i^*j} + c_i$, where c_i is a constant for row i. If we know a column (reference condition), say, column j^*, in the bi-cluster, we can get a new matrix $B(I,J)$ by setting $b_{ij} = a_{ij} - (a_{ij^*} - a_{i^*j^*})$.

OBSERVATION 1. *If $A(I',J')$ is an error-free additive bi-cluster of reference gene i^* in $A(I,J)$, then $B(I,J)$ is an error-free constant bi-cluster for reference gene i^* in $B(I,J)$.*

EXAMPLE: Consider the error-free additive bi-cluster shown in Figure 1b. Let us use the 4th column in the bi-cluster as the reference condition. After the calculation $b_{ij} = a_{ij} - (a_{ij^*} - a_{i^*j^*})$, the bi-cluster becomes an error-free constant bi-cluster with four identical rows: 1.0, 2.0, 5.0, 0.0.

In practice, we do not know the reference condition j^*. Thus, we have to try every column in J as the reference condition. The complete algorithm for finding additive bi-clusters is given in Figure 3. We refer to this algorithm as additive MSBE algorithm.

The Complete Algorithm for Computing Additive Bi-clusters

Input An $n \times m$ matrix $A(I,J)$.
Output A set of additive bi-clusters.

1. **For** every column j^* in J **do**
2. Compute $B(I,J)$ using $b_{ij} = a_{ij} - (a_{ij^*} - a_{i^*j^*})$.
3. **For** each row i^* in I **do**
4. Convert $B(I,J)$ into $S(I,J)$ based on row i^*
5. Use the γ-MSB algorithm to compute the bi-cluster $S(I_A, J_A)$.
6. Use the extension algorithm to get extended bi-cluster $S(I_E, J_E)$ from $S(I_A, J_A)$.
7. Output the bi-cluster $A(I_E, J_E)$.

Fig. 3. The complete algorithm for computing additive bi-clusters (additive MSBE).

The γ-MSB algorithm (Steps 5) runs in $O((n+m)^2)$ time. In Step 1 and Step 3, we have to select $O(nm)$ times the reference gene i^* and the reference condition j^*. Therefore, the total running time of additive MSBE algorithm is $O(nm(n+m)^2)$. Typically, the number of genes n is a few thousands and the number of conditions is about one hundred. Thus, the algorithm is a bit slow to work for real instances. In the next subsection, we develop an randomized algorithm to speed up the algorithm.

2.6 Randomized algorithm

When there is no additional information of the reference gene, we can simply enumerate all genes as the reference genes. For the additive bi-cluster, we can also enumerate all conditions as the reference conditions. In this case, the running time of the algorithms for the constant bi-cluster and the additive bi-cluster are $O(n(n+m)^2)$ and $O(nm(n+m)^2)$, respectively. To accelerate the computing speed, we can randomly select a part of genes as the reference genes.

Consider a $b \times c$ bi-cluster in an $n \times m$ matrix. If we randomly select $\frac{n}{b}$ genes, the expectation of the number of selected genes that are in the $b \times c$ bi-cluster is 1. In practice, due to the existence of error, we can get better result if we use more than one reference gene in the bi-cluster and select the best result. Therefore, when there is no information about reference gene and reference condition, we randomly select a set of rows and a set of columns as reference genes and reference conditions. We call this algorithm Randomized MSBE (RMSBE). In this way, the running time can be dramatically reduced. Experiments show that this approach can give good solutions in practice.

3 RESULTS

We implemented the algorithms in Java. In this section, we will test the programs and compare our programs with some famous exiting programs. The test platform was a desktop PC with P4 2.8G CPU and 512 M memory running Linux operating system.

To evaluate the performances of different methods, we use the measure (match score) similar to the score proposed in Prelić *et al.* (2006). Let M_1, M_2 be two sets of bi-clusters. The match score of M_1 with respect to M_2 is given by

$$S(M_1,M_2) = \frac{1}{|M_1|} \sum_{A(I_1,J_1)\in M_1} \max_{A(I_2,J_2)\in M_2} \frac{|I_1\cap I_2| + |J_1\cap J_2|}{|I_1\cup I_2| + |J_1\cup J_2|}.$$

Let M_{opt} denote the set of implanted bi-clusters and M the set of the output bi-clusters of a bi-clustering algorithm. $S(M_{\text{opt}}, M)$

Table 1. Parameter settings for different bi-clustering methods

Method	Types	Parameter settings
SAMBA	C	$D = 40$, $N_1 = 4$, $N_2 = 4$, $k = 20$, $L = 10$
BiMax	C	minimum number of genes and chips: 4
ISA	C/A	$t_g = 2.0$, $t_c = 2.0$, seeds = 500
CC	C	$\delta = 0.5$, $\alpha = 1.2$
CC	A	$\delta = 0.002$, $\alpha = 1.2$
RMSBE	C/A	$\alpha = 0.4$, $\beta = 0.5$, $\gamma = \gamma_e = 1.2$
OPSM	A	$l = 100$

'C' stands for constant bicluster, 'A' stands for additive bicluster.

represents how well each of the true bi-clusters is discovered by the bi-cluster algorithm.[1]

3.1 Constant bi-cluster

In order to compare our program with other programs for constant bi-cluster data, we follow the approach proposed in Prelić *et al*. (2006). We compare different bi-clustering methods on constant bi-clusters. Here we consider CC (Cheng and Church, 2000), Samba (Tanay *et al*., 2002), ISA (Ihmels *et al*., 2002, 2004) and Bimax (Prelić *et al*., 2006). We downloaded the software BicAt developed by Barkow *et al*. (2006) and EXPANDER developed by Shamir *et al*. (2005). In BicAt, CC, ISA and Bimax are implemented in Java. Samba is an integrated bi-clustering method in EXPANDER.

Gene expression micro-array experiments often generate datasets with multiple missing expression values (Troyanskaya *et al*., 2001). To imitate the missing values, we add noise by replacing some elements in the matrix with random values. There are three variables b, c and δ in the generation of the bi-clusters. b and c are used to control the size of the implanted bi-cluster. δ is the noise level of the bi-cluster. For constant bi-clusters, some bi-clustering methods only consider up-regulated or down-regulated expression values. To compare with these bi-clustering methods, we use bi-clusters with only up-regulated expression values. In detail, the matrix with implanted constant bi-clusters is generated with four steps: (1) generate a 100×100 matrix A such that all elements of A are 0's, (2) generate ten 10×10 bi-clusters such that all elements of the bi-cluster are 1's, (3) implant the ten bi-clusters into A without overlap, (3) replace $\delta \cdot (100 \times 100)$ elements of the matrix with random noise values (0 or 1) . For each test on constant bi-clusters, we generate 10 matrices and consider the average performances of different bi-cluster methods on the matrices.

To accommodate the constant bi-cluster with only up-regulated expression values, we only consider up-regulated values in the computation of similarity scores in RMSBE. The similarity scores for low expression values are always zeros. In the experiment, the noise level ranges from 0 to 0.25. The parameter settings used for different bi-clustering methods are the default settings and are listed in Table 1. The results for RMSBE in Section 3.1 are obtained by trying every row as the reference gene. The results are shown in

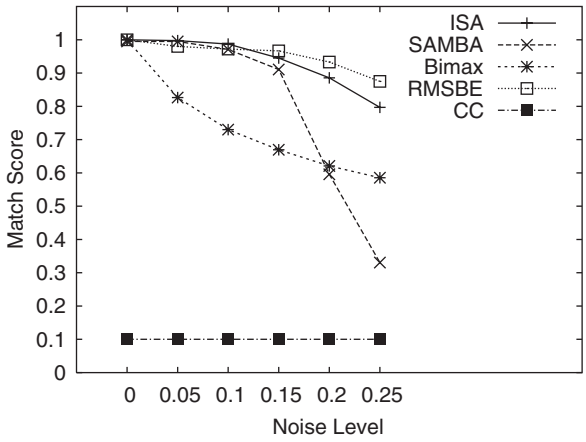

Fig. 4. Results for constant bi-clusters.

Figure 4. In the absent of noise, ISA, Samba, Bimax and RMSBE can always find the implanted bi-clusters correctly. As mentioned in Prelić *et al*. (2006), CC uses the similarity of the selected elements as the bi-clustering criterion. The criterion does not work for the constant bi-cluster with only up-regulated values. CC always outputs the whole matrix as the bi-clustering with CC's parameter $\delta = 0.5$. When the noise level is high, ISA and RMSBE have the best performances. The reference gene set of ISA and the reference gene of RMSBE are the main reasons for them to identify bi-clusters in noisy data. The inclusion-maximal bi-cluster model of Bimax is limited in finding error-free bi-clusters. This model limits its performance on noisy data. The performance of Samba is sensitive to the statistical significance of the bi-clusters. When the noise level is high, the significance of the bi-clusters decreases rapidly. Therefore, the performance of Samba is not good for noisy data. The comparison illustrates the advantage of using the reference gene in our method.

3.2 Additive bi-clusters

Since most of the real datasets in Section 3.3 are cDNA microarray data, we do simulation on cDNA microarray data for additive model. We take the logarithm with base 2 for every cell in the array. This is a standard transformation for microarray analysis. The motivation of the logarithm transformation is that the multiplicative model becomes the additive model. Another reason is that after the transformation, the distribution properties will be better, i.e. the distributions are closer to normal distributions. (Causton *et al*., 2003; Kluger *et al*., 2003).

We randomly generate the values in the 100×100 (background) matrix A such that the data fits the standard normal distribution with the mean of 0 and the SD of 1.[2] To generate an additive $b \times c$ bi-cluster, we first randomly generate the expression values in a reference gene $(a_1, a_2, \ldots, a_c)$ according to the standard normal

[1] $S(M_1, M_2) = \frac{1}{|M_1|}\sum_{A(I_1, J_1)\in M_1} \max_{A(I_2, J_2)\in M_2} \frac{|I_1 \cap I_2|}{|I_1 \cup I_2|}$ is used in Prelić *et al*. (2006). In this paper, we consider both genes and conditions in computing the match score. In the experiments, the two match scores have similar test results. See Supplementary material for the test results using the match score in Prelić *et al*. (2006).

[2] We also test the performances of RMSBE and other biclustering methods on the data fitting the normal distribution with a mean of 0 and SD = 0.5 (some normalized cDNA microarray datasets have this kind of distributions) and the data fitting the normal distribution with a mean of 7 and SD = 1 (some normalized Affymetrix GeneChips datasets have this kind of distributions.). The RMSBE method has similar performances on different types of distributions. See Supplementary material for the test results.

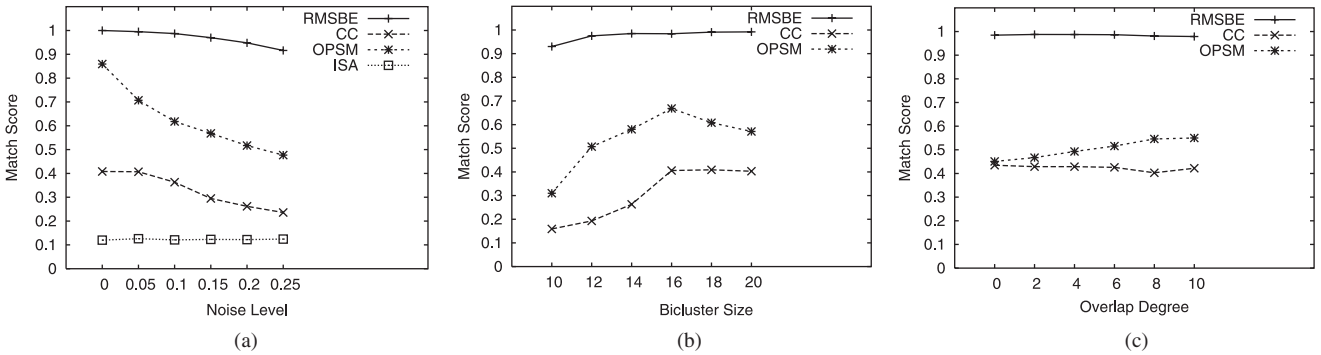

Fig. 5. Results for additive bi-clusters.

distribution. To get a row $(a_{i1}, a_{i2}, \ldots, a_{ic})$ in the additive bi-cluster, we randomly generate a distance d_i (based on the standard normal distribution) and set $a_{ij} = a_j + d_i$ for $j = 1, 2, \ldots, c$. After we get the $b \times c$ additive bi-cluster, we can add some noise by randomly selecting $\delta \cdot (b \times c)$ elements in the bi-cluster and changing the values to a random number (according to the standard normal distribution). Finally, we insert the noisy additive bi-cluster into the 100×100 background matrix A. For each test on additive bi-clusters, 50 matrices are generated. First, let us focus on selecting the parameters.

Parameters selection. We have done some simulations on selecting the parameters. The experiment results are in the Supplementary Material. Based on the simulation results, we find that RMSBE works well for a wide range of parameters settings. We recommend to use the following parameter settings: $\alpha \in [0.2, 0.4]$, $\beta \in [0.0, 0.5]$ and $\gamma \in [\beta + 0.7, \beta + 0.9]$. For the parameter γ_e in the extension algorithm, we use $\gamma_e = \gamma$. Most of results in Section 3.2 for RMSBE are obtained by trying every row as the reference gene and every column as the reference condition if not clearly stated.

Testing additive bi-clusters. Now, we test different programs for additive bi-clusters. The discretization methods used by Samba and Bimax cannot identify the elements in the additive bi-clusters. Without reasonable discretized data, the two methods cannot find the implanted additive bi-clusters. Thus, the two methods are not included in the comparison on additive bi-clusters. In the test on additive bi-clusters, we compared RMSBE with ISA, CC and OPSM implemented in BicAt. We generate the additive bi-clusters of size 15×15 with different noise level δ in [0, 0.25]. The parameter settings of different methods are listed in Table 1. Figure 5a shows that RMSBE has better performance than CC, OPSM and ISA on different noise levels. ISA uses only up-regulated and down-regulated expression values in its bi-clustering method. When an additive bi-clusters contain elements of normal expression levels, ISA may miss some rows and columns of the implanted bi-clusters. When the signal of the implanted bi-cluster is weak comparing with the background noise of the whole matrix, the heuristic methods of CC and OPSM may delete some rows and columns of the implanted bi-cluster in the beginning of the algorithms and miss the deleted rows and columns in the output bi-clusters. For RMSBE, the computation of the similarity scores with the reference gene can filter out many noise and make it easier to find the implanted bi-clusters. Therefore, RMSBE has the best performance in this scenario.

Testing sizes of bi-clusters. Since OPSM and CC work reasonably well for additive bi-clusters, we also compare RMSBE with OPSM and CC on different sizes. In this test, the noise level is $\delta = 0.1$. The sizes of the square additive bi-clusters changes from 10×10 to 20×20. When the sizes of implanted bi-clusters are small, the match scores of OPSM and CC decrease rapidly and RMSBE can still find the implanted bi-clusters (Fig. 5b). From this point of view, RMSBE is more powerful for finding small bi-clusters.

Finding more than one bi-cluster. To test data with more than one bi-cluster, we first generate two $b \times b$ additive bi-clusters with o overlapped rows and columns. o is called the overlap degree. Also, we replace δ fraction of the two bi-clusters with random noise values and implant them into a 100×100 randomly generated matrix. The elements also fit the standard normal distribution. To find more than one bi-cluster in a given matrix, some methods, e.g. CC, need to mask the discovered bi-clusters with random values. Another advantages of the reference gene method (RMSBE) is that it does not need to mask discovered bi-clusters. We test the performance of RMSBE, CC and OPSM on overlapped biclusters by using 20×20 additive bi-clusters with noise level $\delta = 0.1$ and overlap degree o ranging from 0 to 10. The results in Figure 5c show that RMSBE is only marginally affected by the overlap degree of the implanted bi-clusters.

Testing rectangle bi-clusters. To test the extension algorithm for rectangle bi-clusters, we generated additive bi-clusters with different sizes and different noise levels. The sizes of the implanted bi-clusters are from 10×10 square bi-clusters to 30×10 rectangle bi-clusters. The noise level δ is in [0, 0.25]. In the experiment, the parameters of RMSBE are shown in Table 1. The results show that the match scores only slightly decrease when the sizes of the implanted bi-clusters vary from 10×10 to 30×10 (Fig. 6a). RMSBE uses the bi-cluster discovered by MSB algorithm as a core and adds other genes with similar expression patterns into the bi-cluster. With the extension algorithm, RMSBE overcomes its limitation on rectangle bi-clusters.

The accuracy and running time of RMSBE. Here we test the accuracy and running time of the randomized algorithm RMSBE. We generated a 2000×200 matrix (typical size of the gene expression matrix) with an implanted 20×20 additive bi-cluster. We selected $k \times 100$ genes and $k \times 10$ conditions as the reference genes and the reference conditions, where k ranged from 1 to 10. The accuracy rate that the implanted bi-cluster is discovered is shown in Figure 6b and the running time is shown in Figure 6c.

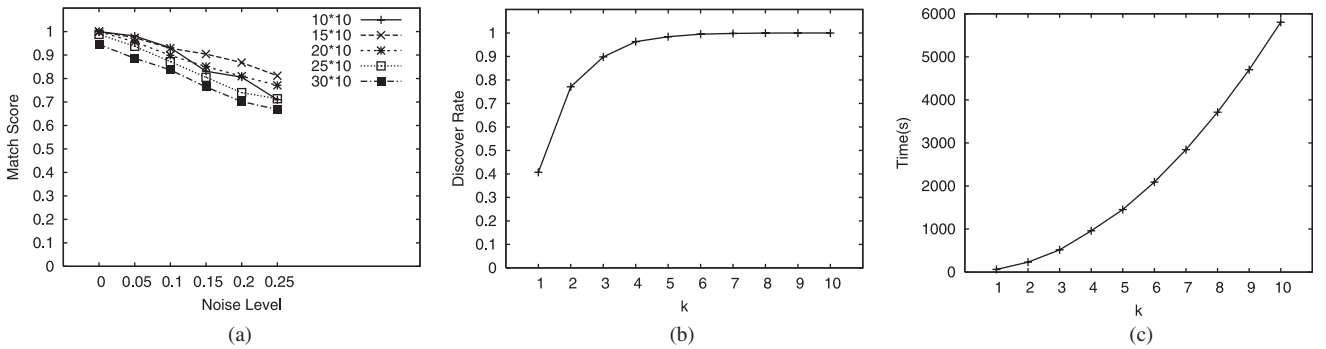

Fig. 6. (**a**) Testing on rectangle biclusters. (**b**) and (**c**) Testing on the randomized algorithm.

From Figure 6b and c, we can see that, when $k = 5$, the discover rate is over 98% and the running time is much shorter than selecting all rows and all columns. Therefore, to obtain good performance in short time, we recommend to set $k = 5$ or 6.

3.3 Real data

Following the method in Prelić *et al*. (2006), we test the programs on real datasets. The discovered bi-clusters are evaluated using Gene Ontology (GO) annotations and protein–protein interaction networks for the first two datasets. The dataset for metabolic pathway maps in Prelić *et al*. (2006) is not included since the dataset is not available. The above two datasets are cNDA data with logarithm transformation. We also test the colon cancer dataset in Alon *et al*. (1999).

Gene Ontology. Similar to the method used by Tanay *et al*. (2002) and Prelić *et al*. (2006), we investigated whether the set of genes discovered by bi-clustering methods shows significant enrichment with respect to a specific GO annotation provided by Gene Ontology Consortium (Gene Ontology Consortium, 2000). We used the web tool FuncAssociate (Berriz *et al*., 2003) to evaluate the discovered bi-clusters. FuncAssociate first uses Fisher's Exact Test to compute the hypergeometric functional score of a gene set, then uses the Westfall and Young procedure (Westfall and Young, 1993) to compute the adjusted significant score of the gene set. The analysis is performed on the gene expression data of *S. cerevisiae* provided by Gasch *et al*. (2000). The dataset contains 2993 genes and 173 conditions. We used parameters $\alpha = 0.4$, $\beta = 0.5$ and $\gamma = \gamma_e = 1.2$ and randomly selected 300 genes and 40 conditions as the reference genes and reference conditions. We also filtered out the bi-clusters with over 25% overlapped elements and output the largest 100 bi-clusters. The running time of RMSBE on this test was 1230 s. The adjusted significant scores (adjusted *P*-values) of the 100 bi-clusters were computed by using FuncAssociate. The significant scores are compared with the results of OPSM, BiMax, ISA, Samba and CC obtained from Figure 3 in Prelić *et al*. (2006). The result is summarized in Figure 7. The result shows that 98% of discovered bi-clusters by RMSBE are statistically significant, $\alpha \leq 0.001\%$. Compared with other methods, RMSBE obtains the best result. The genes with high similarity scores with the reference gene also are highly enriched with the GO biological process category.

Protein–protein interaction network. We also studied the relationship between the discovered bi-clusters with the protein–protein interaction networks. We followed the method proposed by Prelić

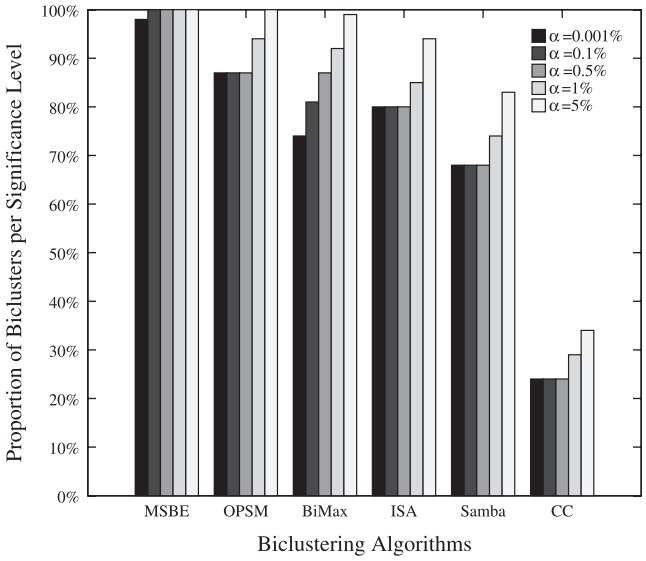

Fig. 7. Proportion of bi-clusters significantly enriched by any GO biological process category (*S.cerevisiae*). α is the adjusted significant scores of the bi-clusters.

et al. (2006). We used the *S.cerevisiae* dataset containing 3665 genes and 173 conditions. The protein–protein networks is obtained from the DIP database (Salwinski *et al*., 2004). For each pair of genes, we can check whether the two genes are connected in the protein–protein networks. If two genes are connected, we are interested in the shortest path between the two genes. We expect that the number of disconnected gene pairs and the average shortest distance between connected gene pairs are smaller for the discovered bi-clusters than for random gene groups. We used the same parameters used in the GO enrichment test and obtained 100 largest bi-clusters. For each discovered bi-cluster, we used Z-test to check whether its proportion of disconnected pairs and average shortest distance are significantly smaller or greater (significance level $\alpha \leq 0.001$) than the expected values for random gene groups. RMSBE finds 50 bi-clusters with significantly smaller disconnectedness degrees and 49 bi-clusters with significantly smaller average distance. RMSBE also finds 17 bi-clusters with significantly greater disconnectedness degrees and 25 bi-clusters with significantly greater average distance. The result is not clear enough to prove the relevance between the discovered bi-clusters and the protein–protein

Table 2. The bi-clusters found in colon cancer dataset

Bi-cluster	Method	#Genes	#Samples	#Tumor	#Normal
B1	XMOTIF	11	15	14	1
B2	XMOTIF	13	18	2	16
B3	RMSBE	27	24	24	0
B4	RMSBE	16	12	0	12

networks. But about half of the discovered bi-clusters really show smaller disconnectedness degree and smaller average shortest distance. As discussed in Prelić *et al*. (2006), the incompleteness of the data and the confidence of the measurement in the protein–protein network may be the reasons for the unclear result.

Colon cancer dataset. Murali and kasif (2003) used a colon cancer dataset originated in Alon *et al*. (1999) to test XMOTIF. The matrix contains 40 colon tumor samples and 22 normal colon samples over about 6500 genes. The dataset is available at http://www.weizmann.ac.il/physics/complex/compphys (Getz *et al*., 2000). In Murali and Kasif (2003), the best two bi-clusters generated by the software XMOTIF are B1 and B2 (Table 2). B1 contains 11 genes and 15 samples. Among the 15 samples, 14 of them are tumor samples and 1 of them is a normal sample. B2 contains 13 genes and 18 samples. Among the 18 samples, 16 of them are normal and 2 of them are tumor. We use $\alpha = 0.4$, $\beta = 0.5$ and $\gamma = \gamma_e = 1.2$, and randomly select 500 genes and all conditions as the reference genes and reference conditions to run our program RMSBE. The best two bi-clusters that we find are B3 and B4 in Table 2. B3 contains 27 genes and 24 samples, where all of the 24 samples are tumor samples. B4 contains 16 genes and 12 samples, where all of the 12 samples are normal. The results show that the RMSBE method can find high quality bi-clusters. Bi-cluster B3 is for tumor samples and B4 is for normal samples.

ACKNOWLEDGEMENTS

The authors thank the referees for their helpful suggestions. The authors also thank Stefan Bleuler for providing test datasets used in Section 3.3. This work is fully supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China [Project No. CityU 1070/02E].

Conflict of Interest: none declared.

REFERENCES

Alon,U. *et al*. (1999) Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Natl Acad. Sci. USA*, **96**, 6745–6750.

Baldi,P. and Hatfield,G.W. (2002) *DNA Microarrays and Gene Expression: From Experiments to Data Analysis and Modeling*. Cambridge University Press, Cambridge.

Barkow,S. *et al*. (2006) BicAT: a biclustering analysis toolbox. *Bioinformatics*, **22**, 1282–1283.

Ben-Dor,A., Chor,B., Karp,R. and Yakhini,Z. (2002) Discovering local structure in gene expression data: the order-preserving submatrix problem. In *Proceedings of the Sixth International Conference on Computational Molecular Biology (RECOMB 2002),* ACM Press, Washington, DC, pp. 49–57.

Berriz,G.F. *et al*. (2003) Charactering gene sets with FuncAssociate. *Bioinformatics*, **19**, 2502–2504.

Causton,H.C., Quackenbush,J. and Brazma,A. (2003) *Microarray Gene Expression Data Analysis : A Beginner's Guide*. Blackwell Publishing, Malden.

Cheng,Y. and Church,G.M. (2000) Biclustering of expression data. In *Proceedings of the 8th International Conference on Intelligent Systems for Molecular (ISMB-00),* AAAI Press, Menlo Park, CA, pp. 93–103.

Gasch,A.P. *et al*. (2000) Genomic expression programs in the response of yeast cells to environmental changes. *Mol. Biol. Cell*, **11**, 4241–4257.

Gene Ontology Consortium (2000), Gene Ontology: tool for the unification of biology. *Nat. Genet*., **25**, 25–29.

Getz,G. *et al*. (2000) Coupled two-way clustering analysis of gene microarray data. *Proc. Natl Acad. Sci. USA*, **97**, 12079–12084.

Ihmels,J. *et al*. (2004) Defining transcription modules using large-scale gene expression data. *Bioinformatics*, **20**, 1993–2003.

Ihmels,J. *et al*. (2002) Revealing modular organization in the yeast transcriptional network. *Nat. Genet*., **31**, 370–377.

Kluger,Y. *et al*. (2003) Spectral biclustering of microarray data: coclustering genes and conditions. *Genome Res*., **13**, 703–716.

Madeira,S.C. and Oliveira,A.L. (2004) Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Trans. Comput. Biol. Bioinform*., **1**, 24–45.

Murali,T.M. and Kasif,S. (2003) Extracting conserved gene expression motifs from gene expression data. In *Proceedings of the 8th Pacific Symposium on Biocomputing* Lihue, Hawaii, pp. 77–88.

Prelić,A. *et al*. (2006) A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics*, **22**, 1122–1129.

Salwinski,L. *et al*. (2004) The database of interacting proteins: 2004 update. *Nucleic Acids Res*., **32**, D449–D451.

Shamir,R. *et al*. (2005) EXPANDER—an integrative program suite for microarray data analysis. *BMC Bioinformatics*, **6**, 232.

Tanay,A. *et al*. (2002) Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, **18** (Suppl. 1), 136–144.

Troyanskaya,O. *et al*. (2001) Missing value estimation method for DNA microarrays. *Bioinformatics*, **17**, 520–525.

Westfall,P.H. and Young,S.S. (1993) *Resampling-Based Multiple Testing*. Wiley, NY.

Yang,J. *et al*. (2002) δ-clusters: capturing subspace correlation in a large data set. In *Proceedings of the 18th International Conference on Data Engineering*, San Jose, pp. 517–528.