A New Framework for Co-clustering of Gene Expression Data*

Shuzhong Zhang[1] Kun Wang[2] Bilian Chen[3] Xiuzhen Huang[4]

[1] Industrial and Systems Engineering Program, University of Minnesota,
Minneapolis, MN 55455, USA. Email: zhangs@umn.edu.
(On leave from Department of Systems Engineering and
Engineering Management, The Chinese University of Hong Kong,
Shatin, Hong Kong. Email: zhang@se.cuhk.edu.hk)

[2] Department of Computer Science, Arkansas State University,
Jonesboro, AR 72467, USA. Email: kun.wang@smail.astate.edu

[3] Department of Systems Engineering and Engineering Management,
The Chinese University of Hong Kong, Shatin, Hong Kong.
Email: blchen@se.cuhk.edu.hk

[4] Department of Computer Science, Arkansas State University,
Jonesboro, AR 72467, USA. Email: xhuang@astate.edu

Abstract. A new framework is proposed to study the co-clustering of gene expression data. This framework is based on a generic tensor optimization model and an optimization method termed *Maximum Block Improvement* (MBI) recently developed in [3]. Not only can this framework be applied for co-clustering gene expression data with genes expressed at different conditions represented in 2D matrices, but it can also be readily applied for co-clustering more complex high-dimensional gene expression data with genes expressed at different tissues, different development stages, different time points, different stimulations, etc. Moreover, the new framework is so flexible that it poses no difficulty at all to incorporate a variety of clustering quality measurements. In this paper, we demonstrate the effectiveness of this new approach by providing the details of one specific implementation of the algorithm, and presenting the experimental testing on microarray gene expression datasets. Our results show that the new algorithm is very efficient and it performs well for identifying patterns in gene expression datasets.

1 Introduction

Microarray and next-generation sequencing (also, high-throughput sequencing) technologies produce huge amount of datasets of genome-wide gene expression at different tissues, different development stages, different time points, and different stimulations. These datasets could significantly facilitate and benefit biological

* This research is partially supported by NIH Grant # P20 RR-16460 from the IDeA Networks of Biomedical Research Excellence (INBRE) Program of the National Center for Research Resources. The corresponding author' email address is: xhuang@astate.edu.

hypothesis testing and discovery. However, the availability of these gene expression datasets at the same time brings the challenge of how to transform the large amount of gene expression data to information meaningful for biologists and life-scientists. Especially this imposes increasing demands for efficient computational models and approaches for processing and analyzing these data.

Clustering, as an effective approach, is usually applied to partition gene expression data into groups, where each group aggregates genes with similar expression levels. All the classical clustering algorithms are focused on clustering genes into a number of groups based on their similar expression on all the considered conditions.

Cheng and Church [4] introduced the concept of co-cluster gene expression data and developed an effective measure of the co-clusters based on the mean square residue and a greedy node-deletion algorithm. Their algorithm could cluster genes and conditions simultaneously and thus could discover the similar expression of a certain group of genes on a certain group of conditions and vice versa. Later many different co-clustering algorithms were developed. For example, the authors in [5] formulated the objective functions based on minimizing two measures of squared residue that are similar to those used by Cheng and Church [4] and Hartigan [6]. Their iterative algorithm could directly minimize the squared residues and find $k * l$ co-clusters simultaneously as opposed to finding a single co-cluster at a time like Cheng and Church. Readers may refer to [11, 7, 5] for the ideas of other co-clustering algorithms.

In this paper we propose a new framework to study the co-clustering of gene expression data. This new framework is based on a generic tensor optimization model and a method termed *Maximum Block Improvement* (MBI). This framework not only can be used for co-clustering of gene expression data with genes expressed at different conditions (genes × conditions) represented in 2D matrices, but also it can be readily applied for co-clustering of gene expression data in 3D, 4D, 5D with genes expressed at different tissues, different development stages, different time points, different stimulations, and so on and so forth (e.g., genes×tissues×development stages×time points×stimulations) and even more complex high-dimensional matrices. Moreover, this framework is flexible enough to incorporate different objective functions. We demonstrate this new framework by providing the details of the algorithm for one model with one specific objective function under the framework, the implementation of the algorithm and the experimental testing on microarray gene expression datasets. Our algorithm turns out to be very efficient (which runs for only a few minutes on a regular PC for large gene expression datasets) and performs well for identifying patterns in microarray data sets compared with other approaches (refer to the section of experimental results).

The reminder of the paper is organized as follows. Section 2 presents the new generic co-clustering framework. Section 3 describes the algorithm for one specific 2D gene expression co-clustering model. Section 4 presents experimental testing results on gene expression datasets. Section 5 concludes the paper.

2 A new generic framework for co-clustering

In this section we first present our model for the co-clustering problem based on tensor optimization and then give a generic algorithm for high-dimensional gene expression data co-clustering.

2.1 Background of tensor operations

Readers are referred to [9] for different tensor operations. We will need in the following the operation *mode product* between a tensor X and a matrix P. Suppose that $X \in \Re^{p_1 \times p_2 \times \cdots \times p_d}$ is an d-dimensional tensor and $P \in \Re^{p_i \times m}$ is a 2D matrix. Then, $X \times_i P$ is a tensor in $\Re^{p_1 \times p_2 \times \cdots \times p_{i-1} \times m \times p_{i+1} \times \cdots \times p_d}$, whose $(j_1, j_2, \cdots, j_{i-1}, j_i, j_{i+1}, \cdots, j_d)$-th component is defined by

$$(X \times_i P)_{j_1, j_2, \cdots, j_{i-1}, j_i, j_{i+1}, \cdots, j_d} = \sum_{\ell=1}^{p_i} X_{j_1, j_2, \cdots, j_{i-1}, \ell, j_{i+1}, \cdots, j_d} P_{\ell, j_i}.$$

The mode product is communicative, i.e.,

$$X \times_i P \times_j Q = X \times_j Q \times_i P.$$

2.2 The optimization model of the co-clustering problem

The co-clustering problem is described as follows. Suppose that $A \in \Re^{n_1 \times n_2 \times \cdots \times n_d}$ is a d-dimensional tensor. Let $I_j = \{1, 2, \cdots, n_j\}$ be the set of indices on the j-th dimension, $j = 1, 2, ..., d$. We wish to find a p_j-partition of the index set I_j, say $I_j = I_1^j \cup I_2^j \cup \cdots \cup I_{p_j}^j$, where $j = 1, 2, ..., d$, in such a way that each of the *sub-tensor* $A_{I_{i_1}^1 \times I_{i_2}^2 \times \cdots \times I_{i_d}^d}$ is as tightly packed up as possible, where $1 \leq i_j \leq n_j$ and $j = 1, 2, ..., d$.

Suppose that $X \in \Re^{p_1 \times p_2 \times \cdots \times p_d}$ is the tensor for the co-cluster values. Let $X_{j_1, j_2, \cdots, j_{i-1}, j_i, j_{i+1}, \cdots, j_d}$ be the value of the co-cluster $(j_1, j_2, \cdots, j_{i-1}, j_i, j_{i+1}, \cdots, j_d)$ with $1 \leq j_i \leq p_i$, $i = 1, 2, ..., d$.

Next, we define a row-to-column assignment matrix $Y^j \in \Re^{n_j \times p_j}$ for the indices for the j-th array of tensor A, with:

$$Y_{ik}^j = \begin{cases} 1, \text{ if } i \text{ is assigned to the } k\text{-th partition } I_k^j; \\ 0, \text{ otherwise.} \end{cases}$$

Then, we introduce a *proximity* measure $f(s) : \Re \to \Re_+$, with the property that $f(s) \geq 0$ for all $s \in \Re$ and $f(s) = 0$ if and only if $s = 0$. The co-clustering problem can be formulated as

$$(CC) \min \sum_{j_1=1}^{n_1} \sum_{j_2=1}^{n_2} \cdots \sum_{j_d=1}^{n_d} f\left(A_{j_1, j_2, \cdots, j_d} - (X \times_1 Y^1 \times_2 Y^2 \times_3 \cdots \times_d Y^d)_{j_1, j_2, \cdots, j_d}\right)$$
$$\text{s.t. } X \in \Re^{p_1 \times p_2 \times \cdots \times p_d},\ Y^j \in \Re^{n_j \times p_j} \text{ is an assignment matrix}, j = 1, 2, ..., d.$$

A variety of proximity measures could be considered. For instance, if $f(s) = s^2$, then (CC) can be written as

$$
\begin{aligned}
(P_1)\ \min\ & \left\| A - X \times_1 Y^1 \times_2 Y^2 \times_3 \cdots \times_d Y^d \right\|_F \\
\text{s.t.}\ \ & X \in \Re^{p_1 \times p_2 \times \cdots \times p_d},\ Y^j \in \Re^{n_j \times p_j} \text{ is an assignment matrix, } j = 1, 2, ..., d.
\end{aligned}
$$

If $f(s) = |s|$ then (CC) can be written as

$$
\begin{aligned}
(P_2)\ \min\ & \sum_{j_1=1}^{n_1} \sum_{j_2=1}^{n_2} \cdots \sum_{j_d=1}^{n_d} \left| A_{j_1, j_2, \cdots, j_d} - (X \times_1 Y^1 \times_2 Y^2 \times_3 \cdots \times_d Y^d)_{j_1, j_2, \cdots, j_d} \right| \\
\text{s.t.}\ \ & X \in \Re^{p_1 \times p_2 \times \cdots \times p_d},\ Y^j \in \Re^{n_j \times p_j} \text{ is an assignment matrix, } j = 1, 2, ..., d.
\end{aligned}
$$

A third possible formulation can be

$$
\begin{aligned}
(P_3)\ \min\ & \max_{1 \le j_i \le n_i;\, i=1,2,...,d} \left| A_{j_1, j_2, \cdots, j_d} - (X \times_1 Y^1 \times_2 Y^2 \times_3 \cdots \times_d Y^d)_{j_1, j_2, \cdots, j_d} \right| \\
\text{s.t.}\ \ & X \in \Re^{p_1 \times p_2 \times \cdots \times p_d},\ Y^j \in \Re^{n_j \times p_j} \text{ is an assignment matrix, } j = 1, 2, ..., d.
\end{aligned}
$$

2.3 A generic algorithm for co-clustering

In this section we provide an algorithm for the (CC) model of the co-clustering problem. The algorithm is based on a recent work in mathematical optimization ([3]), where the authors considered a generic optimization model in the form of

$$
\begin{aligned}
(G)\ \max\ & f(x^1, x^2, \cdots, x^d) \\
\text{s.t.}\ \ & x^i \in S^i \subseteq \Re^{n_i},\ i = 1, 2, ..., d,
\end{aligned}
$$

where $f : \Re^{n_1 + \cdots + n_d} \to \Re$ is a general continuous function, and S^i is a general set, $i = 1, 2, ..., d$. They proposed a new method termed Maximum Block Improvement (MBI) for solving the optimization problem (G).

Note that in [3], the authors proved that the Maximum Block Improvement Method method guarantees to converge to a stationary point:

Theorem 1. *([3]) If S^i is compact, $i = 1, 2, ..., d$, then any cluster point of the iterates $(x^1_k, x^2_k, \cdots, x^d_k)$, say $(x^1_*, x^2_*, \cdots, x^d_*)$, will be a stationary point for (G); i.e.,*

$$
x^i_* = \arg \max_{x^i \in S^i} f(x^1_*, \cdots, x^{i-1}_*, x^i, x^{i+1}_*, \cdots, x^d_*), \ \ \textit{for } i = 1, 2, ..., d.
$$

We can see that all our formulations, (P_1), (P_2) and (P_3), are in the format of (G), which are suitable for the application of the MBI method. Refer to Figure 1 for our generic algorithm for the co-clustering problem based on the MBI method.

The model contains the block variables X, Y^1, Y^2, ..., Y^d. For the fixed Y^j variables, $j = 1, 2, ..., d$, the search of X becomes:

- In the case of (P_1), the problem is a least square problem;
- In the case of (P_2) or (P_3), the problems are linear programming.

To appreciate the computational complexity of the models under consideration, we remark here that even if the X block variable is fixed, to search for the *two* joint optimal assignments of, say, Y^1 and Y^2, while all other Y's are fixed, is already NP-hard. (We omit the proof here due to the space limit).

Generic co-clustering algorithm

Input: $A \in \Re^{n_1 \times n_2 \times \cdots \times n_d}$ is an d-dimensional tensor. Parameters k_1, k_2 and k_d, are all positive integers, $0 < k_i \leq n_i$, $1 \leq i \leq d$.
Output: $k_1 \times k_2 \times \cdots \times k_d$ co-clusters of A.

Main Variables:
A non-negative integer k as the loop counter;
A $k_1 \times k_2 \cdots \times k_d$-tensor X with each entry a real number as the artificial central point of one of the co-clusters;
A $n_i \times k_i$-matrix Y_i as the assignment matrix with $\{0, 1\}$ as the value of each entry, $1 \leq i \leq d$.

begin

0 *(Initialization).* $Y^0 = X$; Choose a feasible solution $(Y_0^0, Y_0^1, Y_0^2, \cdots, Y_0^d)$ and compute the initial objective value $v_0 := f(Y_0^0, Y_0^1, Y_0^2, \cdots, Y_0^d)$. Set the loop counter $k := 0$.

1 *(Block Improvement).* For each $i = 0, 1, 2, \ldots, d$, solve

$$\begin{array}{rl} (G_i) & \max\ f(Y_k^0, Y_k^1, \cdots, Y_k^{i-1}, Y^i, Y_k^{i+1}, \cdots, Y_k^d) \\ & \text{s.t.}\quad Y^i \in \Re^{n_j \times p_j} \text{ is an assignment matrix,} \end{array}$$

and let

$$\begin{aligned} y_{k+1}^i &:= \arg\max f(Y_k^0, Y_k^1, \cdots, Y_k^{i-1}, Y^i, Y_k^{i+1}, \cdots, Y_k^d) \\ w_{k+1}^i &:= f(Y_k^0, Y_k^1, \cdots, Y_k^{i-1}, y_{k+1}^i, Y_k^{i+1}, \cdots, Y_k^d). \end{aligned}$$

2 *(Maximum Improvement).* Let $w_{k+1} := \max_{1 \leq i \leq d} w_{k+1}^i$ and $i^* = \arg\max_{1 \leq i \leq d} w_{k+1}^i$. Let

$$\begin{aligned} Y_{k+1}^i &:= Y_k^i,\ \forall\, i \in \{0, 1, 2, \cdots, d\} \backslash \{i^*\} \\ Y_{k+1}^{i^*} &:= y_{k+1}^{i^*} \\ v_{k+1} &: := w_{k+1}. \end{aligned}$$

3 *(Stopping Criterion).* If $|v_{k+1} - v_k| < \epsilon$, go to Step 4. Otherwise, set $k := k + 1$, and go to Step 1.

4 *(Outputting Co-clusters).* According to the assignment matrices $Y_{k+1}^1, Y_{k+1}^2, \cdots, Y_{k+1}^d$, print the $k_1 \times k_2 \times \cdots \times k_d$ co-clusters of A.

end

Fig. 1. Algorithm Based on the MBI Method in [3].

3 Algorithm for co-clustering 2D matrix data

We have implemented the algorithm for co-clustering gene expression data in 2D-matrices when the (P_1) formulation is used. Given a 2D-matrix A with m rows and n columns, which represents the gene expressions of m different genes under n different conditions. We apply our co-clustering algorithm to partition the genes and conditions at the same time to get $k_1 \times k_2$ submatrices, where k_1 is the number of partitions of the m genes and k_2 is the number of partitions of the n conditions. Refer to Figure 2 for the details of our algorithm.

4 Experimental results

We use two microarray datasets to test our algorithm and make comparisons with other clustering and co-clustering methods. The first dataset is the gene expression of a yeast cell cycle dataset with 2884 genes and 17 conditions, where the expression values are in the range 0 to 595. The second dataset is the gene expression of a human B-cell lymphoma dataset with 4026 genes and 96 conditions, where the values are in the range -749 and 642. The detailed information about the datasets could be found in Cheng and Church [4], Tavazoie et al. [13] and Alizadeh et al. [2].

4.1 Implementation details and some discussions

Our algorithm is implemented using C++. The experimental testing is performed on a regular PC (configuration: processor: Pentium dual-core CPU, T4200 @ 2.00GHz; memory: 3GB; operating system: 64-bit windows 7; compiler: Microsoft Visual C++ 2010). The figures are generated using MATLAB R2010a.

We tested our algorithm using different initial values of the three matrices X, Y_1 and Y_2 (refer to Figure 2). The setup of the initial values of the three matrices includes using random values for the three matrices, using subsets of values in A to initialize X, limiting the number of 1s to be one in each row of matrices Y_1 and Y_2, and using the values of the matrices Y_1 and Y_2 to calculate the values of the matrix X. We found out that the initial values of the three matrices will not significantly affect the convergence of our algorithm (refer to Figure 3 for the final objective function values and the running times over 50 runs for the yeast dataset to generate 30×3 co-clusters).

We also tested our algorithm for different numbers of partitions of the rows and the columns, that is, different values of k_1 and k_2. For example, when $k_1 = 30$ and $k_2 = 3$, our program generates the co-clusters of the yeast cell dataset in 40.252 seconds with the final objective function value -7386.75, and when $k_1 = 100$ and $k_2 = 5$, our program generates the co-clusters of the yeast cell dataset in 90.138 seconds with the final objective function value -6737.86. The running time of our algorithm is comparable to the running time of the algorithms developed in [5].

Algorithm for 2D-matrix co-clustering based on the P1 model

Input: A 2D-matrix A with m rows and n columns. Two parameters k_1 and k_2, where k_1 and k_2 are both positive integers.
Output: $(k_1 \times k_2)$ co-clusters of the matrix A, where k_1 is the number of partitions of the m rows and k_2 is the number of partitions of the n columns.

Main Variables:
A non-negative integer k as the loop counter;
A $k_1 \times k_2$ matrix X with each entry a real number as the artificial central point of one of the $k_1 * k_2$ co-clusters of the matrix A;
A $m \times k_1$ matrix Y_1 as the row assignment matrix with $\{0,1\}$ as the value of each entry; and
A $n \times k_2$ matrix Y_2 as the column assignment matrix with $\{0,1\}$ as the value of each entry.

```
begin
  0 (Initialization). Set the loop counter $k := 0$. Randomly set the initial values of the three matrices $X^k$, $Y_1^k$ and $Y_2^k$ and compute the initial objective value $v_0 := \max - \|A - X \times_1 Y_1 \times_2 Y_2\|_F$.

  1 (Block Improvement).
    1.1 Based on the values in matrices $X^k$ and $Y_1^k$, get the optimal column assignment matrix $Y_2'$ and compute the objective value $v_{Y_2} := \max - \left\|A - X^k \times_1 Y_1^k \times_2 Y_2'\right\|_F$;

    1.2 Based on the values in matrices $X^k$ and $Y_2^k$, get the optimal row assignment matrix $Y_1'$ and compute the objective value $v_{Y_1} := \max - \left\|A - X^k \times_1 Y_1' \times_2 Y_2^k\right\|_F$;

    1.3 Based on the values in matrices $Y_1^k$ and $Y_2^k$, get the optimal matrix $X'$ and compute the objective value $v_X := \max - \left\|A - X' \times_1 Y_1^k \times_2 Y_2^k\right\|_F$.

  2 (Maximum Improvement). $v_{k+1} := \max\{v_{Y_2}, v_{Y_1}, v_x\}$;
    If $v_{k+1} = v_{Y_2}$ then update $Y_2$:
    $X^{k+1} = X^k$, $Y_1^{k+1} = Y_1^k$, and $Y_2^{k+1} = Y_2'$;
    If $v_{k+1} = v_{Y_1}$ then update $Y_1$:
    $X^{k+1} = X^k$, $Y_1^{k+1} = Y_1'$, and $Y_2^{k+1} = Y_2^k$;
    If $v_{k+1} = v_X$ then update $X$:
    $X^{k+1} = X'$, $Y_1^{k+1} = Y_1^k$, and $Y_2^{k+1} = Y_2^k$;

  3 (Stopping Criterion). If $|v_{k+1} - v_k| < \epsilon$, go to Step 4. Otherwise, set $k := k + 1$, and go to Step 1.
  4 (Outputting Co-clusters). According to the assignment matrices $Y_{k+1}^1, Y_{k+1}^2$, print the $k_1 \times k_2$ co-clusters of $A$.
end
```

Fig. 2. Algorithm for 2D-matrix Co-clustering.

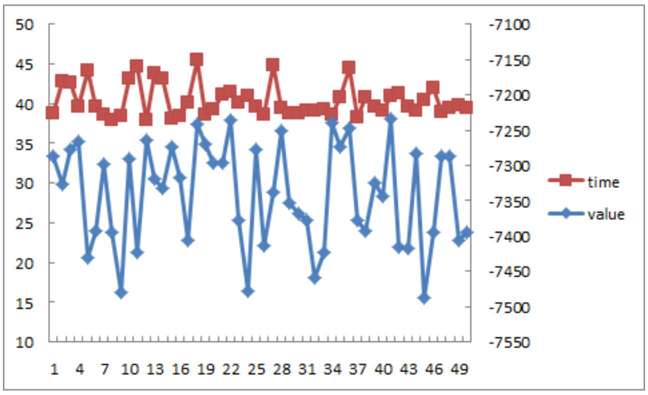

Fig. 3. The final objective function values (the right axis) and the running time (the left axis, in seconds) of 50 runs of our algorithm with random initial values of the three matrices X, Y_1 and Y_2 on the yeast dataset to generate 30×3 co-clusters.

Refer to Figure 4 for the objective function value versus iteration of our algorithm on the yeast cell dataset and the human lymphoma dataset. The average initial and final objective function values over 20 runs for the yeast dataset to generate 30×3 co-clusters are -25818.1 and -7323.42. The average initial and final objective function values over 20 runs for the human lymphoma dataset to generate 150×7 co-clusters are -143958 and -119766. There are 100 iterations of our implemented algorithm. We can see that our algorithm converges rapidly.

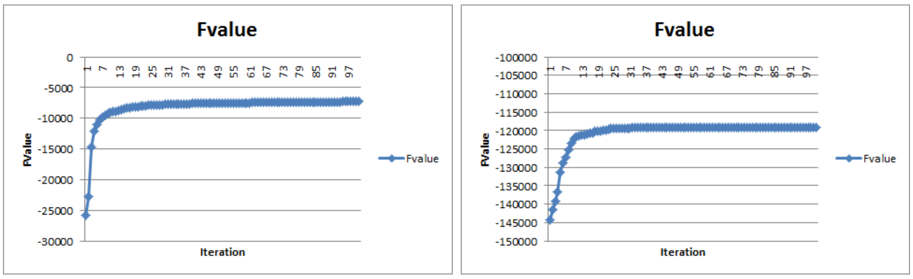

Fig. 4. The figure on the left shows the objective function value vs. iteration of our algorithm on the yeast dataset to generate 30×3 co-clusters. The figure on the right shows the objective function value vs. iteration of our algorithm on the human dataset to generate 150×7 co-clusters.

4.2 Testing results using microarray datasets

In the following we present some exemplary co-clusters identified by our algorithm. We compare the co-clusters with those identified by other approaches. For all the figures presented here, the x-axis represents the different number of conditions and the y-axis represents the values of the gene expression level. (Due to the space limit, some detailed testing results and identified co-clusters are not shown here).

Figure 5 shows four co-clusters of the yeast cell dataset generated when the two parameters $k_1 = 20$ and $k_2 = 3$.

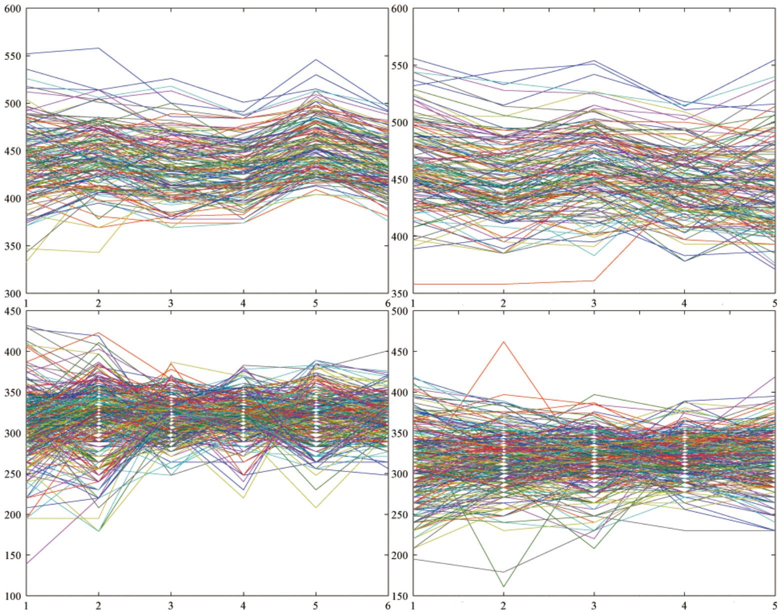

Fig. 5. Four co-clusters of the yeast cell dataset generated when the two parameters $k_1 = 20$ and $k_2 = 3$. The two co-clusters in the same row contain the same sets of genes but in two different sets of conditions, and the two co-clusters in the same column show two different groups of genes on the same set of conditions. Each of the four co-clusters from top-left to bottom-right has the following (number of genes, [list of conditions]) respectively (148, [condition 0, 1, 5, 8, 11, 12]), (148, [condition 2, 3, 4, 6, 7]), (292, [condition 0, 1, 5, 8, 11, 12]), and (292, [condition 2, 3, 4, 6, 7]).

We can see from the co-clusters shown in Figure 5, 6 and other generated co-clusters that our algorithm can effectively identify groups of genes and groups of conditions that exhibit similar expression patterns. It can discover the same subset of genes that have different expression levels over different subsets of conditions, and can also discover different subsets of genes that have different expression levels over the same subset of conditions.

The four co-clusters in Figure 5 are closely related to the clusters of Tavazoie *et al.* [13], where the classical k-means clustering algorithm was applied and the yeast cell cycle gene expression dataset was clustered into 30 clusters. The bottom two co-clusters are mainly related to their clusters 2, 3, 4, 6 and 18. The top two co-clusters are mainly related to their cluster 1. This shows that the same group of genes have different expression patterns over different subsets of conditions. This also shows that one or more than one co-clusters could correspond to one cluster of Tavazoie *et al.* [13].

We use the mean square residue score developed in [4] to evaluate the co-clusters generated by our algorithm. We identify 12 co-clusters with the best mean square residue scores of the yeast cell dataset when $k_1 = 30$ and $k_2 = 3$. The list of the scores are 168.05, 182.04, 215.69, 335.72, 365.01, 378.37, 408.98, 410.03, 413.08, 416.63, 420.37, and 421.49. All the 12 co-clusters have the mean square residue scores less than 450. They are meaningful co-clusters.

We conduct similar experimental testing on the human lymphoma dataset. Figure 6 shows four exemplary co-clusters of the dataset generated when the two parameters $k_1 = 150$ and $k_2 = 7$.

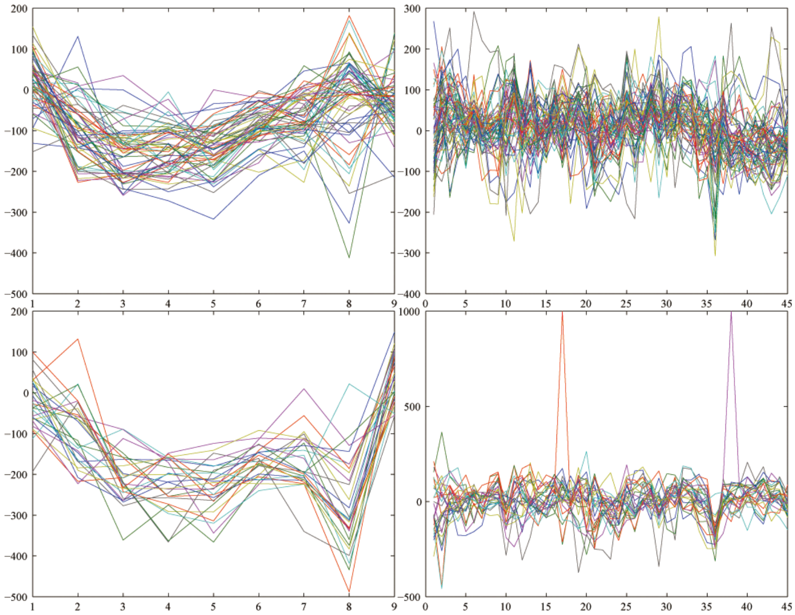

Fig. 6. Four co-clusters of human cell dataset generated when the two parameters $k_1 = 150$ and $k_2 = 7$. Note that two co-clusters in the same row contain the same sets of genes but in different sets of conditions, and the two co-clusters in the same column show two different groups of genes on the same set of conditions. Each of the four co-clusters has the following (number of genes, number of conditions): (57, 9), (57,45), (27,9), and (27,45).

4.3 Testing using 3D synthesis dataset

We test our algorithm using the 3D synthetic dataset from [12] which has six files with each file containing 1,000 genes measured over 10 conditions with 6 time-points for each condition. The co-clusters in Figure 7 show clear coherent patterns of the 3D dataset.

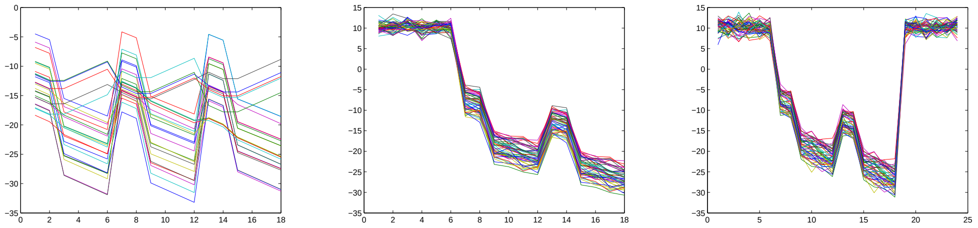

Fig. 7. Co-clusters of the 3D dataset generated when the three parameters $k_1 = 10$, $k_2 = 1$ and $k_3 = 3$. Each curve corresponds to the expression of one gene. The x-axis represents the different number of time points with every 6 time-points in one condition, while the y-axis represents the values of the gene expression level.

5 Summary and future works

We have developed a new framework for co-clustering gene expression data, which includes an optimization model and a generic algorithm for the co-clustering problem. We implemented and tested our algorithm on two 2D microarray datasets and one 3D synthesis dataset.

In the near future, we will extend our algorithm to handle gene expression datasets in high-dimensional tensors, such as genes expressed at different tissues, different development stages, different time points, different stimulations. We will study and test the co-clustering model for identifying scaling and shifting patterns [1] and overlapped co-clusters [4]. We are currently conducting the testing of our models for analyzing different microarray and next-generation sequencing datasets from real-life biological experiments. It will also be very useful to consider pre-processing experimental data such as removing trivial co-clusters as in [4], post-processing the identified co-clusters, and incorporating biological constraints into the co-clustering model. To extract meaningful information for biologists and life-scientists out of the vast experimental gene expression datasets is a hugely challenging task. The marvelous prospect of its success, however, may arguably justify the toil in the process.

References

1. Aguilar-Ruiz JS, Shifting and scaling patterns from gene expression data. *Bioinformatics*, 21(20):3840-3845, Epub 2005.
2. Alizadeh AA, Eisen MB, Davis RE, Ma C, Lossos IS, Rosenwald A, Boldrick JC, Sabet H, Tran T, Yu X, Powell JI, Yang L, Marti GE, Moore T, Hudson J Jr, Lu L, Lewis DB, Tibshirani R, Sherlock G, Chan WC, Greiner TC, Weisenburger DD, Armitage JO, Warnke R, Levy R, Wilson W, Grever MR, Byrd JC, Botstein D, Brown PO, Staudt LM, Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, 403(6769):503-11, 2000.
3. Chen B, He S, Li Z, and Zhang S, Maximum block improvement and polynomial optimization. Submitted for publication, 2011.
4. Cheng Y, and Church GM, Biclustering of expression data. *Proc Int Conf Intell Syst Mol Biol.*, 8:93-103, 2000.
5. Cho H, Dhillon IS, Guan Y, and Sra S, Minimum sum-squared residue co-clustering of gene expression data, *Proceedings of The fourth SIAM International Conference on Data Mining*, 114-125, 2004.
6. Hartigan JA, Direct clustering of a data matrix, *Journal of the American Statistical Association*, 67(337):123-129, 1972.
7. Hochreiter S, Bodenhofer U, Heusel M, Mayr A, Mitterecker A, Kasim A, Khamiakova T, Sanden SV, Lin D, Talloen W, Bijnens L, Ghlmann HWH, Shkedy Z, Clevert D, FABIA: factor analysis for bicluster acquisition, *Bioinformatics*, 26(12):1520-1527, 2010.
8. Kilian J, Whitehead D, Horak J, Wanke D, Weinl S, Batistic O, D'Angelo C, Bornberg-Bauer E, Kudla J, Harter K., The AtGenExpress global stress expression data set: protocols, evaluation and model data analysis of UV-B light, drought and cold stress responses, *The Plant Journal*, 2:347-363, 2007.
9. Kolda TG and Bader BW, Tensor decompositions and applications, *SIAM Review*, 51(3):455-500, 2009.
10. Jegelka S, Sra S, Banerjee A, Approximation algorithms for tensor clustering, *Proceedings of The 20th International Conference on Algorithmic Learning Theory*, 68-383, Springer-Verlag, 2009.
11. Madeira SC, Oliveira AL, Biclustering algorithms for biological data analysis: a survey, *IEEE/ACM Trans. Comput. Biology Bioinform.*, 1(1):24-45, 2004.
12. Supper J, Strauch M, Wanke D, Harter K, Zell A, EDISA: extracting biclusters from multiple time-series of gene expression profiles, *BMC Bioinformatics*, 8:334-347, 2007.
13. Tavazoie S, Hughes JD, Campbell MJ, Cho RJ, Church GM., Systematic determination of genetic network architecture. *Nat Genet.*, 22(3):281-5, 1999.