Discovering α–Patterns from Gene Expression Data

Domingo S. Rodríguez Baena[1], Norberto Diaz Diaz[1],
Jesús S. Aguilar Ruiz[1], and Isabel Nepomuceno Chamorro[2]

[1] Pablo de Olavide University, Seville, Spain
dsrodbae@upo.es, ndiaz@upo.es, diresinf@upo.es
[2] Seville University, Seville,Spain
isabel@lsi.us.es

Abstract. The biclustering techniques have the purpose of finding subsets of genes that show similar activity patterns under a subset of conditions. In this paper we characterize a specific type of pattern, that we have called α–pattern, and present an approach that consists in a new biclustering algorithm specifically designed to find α–patterns, in which the gene expression values evolve across the experimental conditions showing a similar behavior inside a band that ranges from 0 up to a pre–defined threshold called α. The α value guarantees the co–expression among genes. We have tested our method on the *Yeast* dataset and compared the results to the biclustering algorithms of Cheng & Church (2000) and Aguilar & Divina (2005). Results show that the algorithm finds interesting biclusters, grouping genes with similar behaviors and maintaining a very low mean squared residue.

1 Introduction

The DNA microarray technology represents a great opportunity of studying the genomic information as a whole, so we can analyze the relations among thousands of genes simultaneously. The experiments carried out on genes under different conditions produce the expression levels of their transcribed mRNA and this information is stored in DNA chips. The analysis of gene expression data on these biochips is an important tool used in genomic investigations which main objectives range from the study of the functionality of specific genes and their participation in biological process to the reconstruction of diseases conditions and their subsequent prognosis. This huge amount of data has attracted the attention of a lot of researchers because extracting useful information from it represents a big challenge. To deal with it, different statistical and data mining techniques have been applied. Clustering is the most popular approach used in this investigation area [1]. This technique is applied to gene expression data for grouping genes according to their expressions levels under all the experimental conditions, or for creating sets of conditions based on the expression of the genes in study. That is, clustering works only with one dimension: genes or conditions. However, a group of genes can show co–expression under a certain group of conditions, but behave independently under others [2]. Thus, the gene expression data need to be analyzed taking into account the two dimensions at the same time.

A *bicluster* is a subset of genes that show similar activity patterns under a subset of conditions. The research on biclustering started in 1972 with Hartigan's work [3]. Hartigan's algorithm, named *direct clustering*, divides the data matrix into a certain number of sub–matrices with the minimum variance. In that approach, the perfect bicluster was the sub–matrix formed by constant values, i.e., with variance equal to zero. Another way of searching biclusters is to measure the coherence between their genes and conditions. Cheng & Church [4] introduced a measure, the *mean squared residue* (MSR), that computes the similarity among the expression values within the bicluster. Many researchers have based their works on the ideas of Cheng & Church, trying to improve the method and the results. For instance, Cheng & Church replaced the missing values on datasets by random numbers. Yang et. al [5] [6] solved the problem caused by these random fillings considering only the valid values. As a result of this approach an algorithm named FLOC (Flexible Overlapped biClustering) was designed.

Other alternatives in the searching for biclusters have been studied in the last years. Lazzeroni et. al [7] present the *plaid models*. With these models the data matrix is described as a linear function of layers corresponding to its biclusters. Shamir et al. [8] propose a new method to obtain biclusters based on a combination of graph theoretical and statistical modelling of data. In a recent work [9], a generalization of OPSM (Order–Preserving Submatrix) model, introduced by Ben-Dor et al. [10], is presented. The OPSM model is based on the search of biclusters in which a subset of genes induce a similar linear ordering along a subset of conditions. Some techniques search for structures in data matrix to find biclusters: Gerstein et. al [11] create a method for clustering genes and conditions simultaneously based on the search of "checkerboard" patterns in matrices of gene expression data. In the research of Aguilar et. al [12] an evolutionary technique, based on the search of biclusters following a sequential covering strategy and measuring the mean squared residue, is used.

In this work we propose an algorithm to obtain a specific type of bicluster, that we have called α–pattern, with the maximum number of genes and in which the absolute value of the difference between two expression values of any pair of genes under the same condition is not greater than a threshold, α.

The paper is organized as follows: in Section 2 the new type of pattern is formally characterized and the definitions related to the biclustering method are presented; the algorithm is shown in Section 3; in Section 4, we describe the method used and discuss the experimental results, comparing the quality of those generated by Cheng & Church's and Aguilar & Divina's algorithms; finally, the most interesting conclusions are summarized in Section 5.

2 Definitions

The gene expression data are arranged in matrices. A matrix is defined as a triple $M = (G, C, \ell)$, where G and C are two finite sets referred to as *the set of genes* and *the set of experimental conditions* respectively, and $\ell : G \times C \longrightarrow \Re$ is the *level* function. We will denote the real number $\ell(g, c)$ by $\langle g, c \rangle$, and represents the level of expression of the gene g under the condition c.

Definition 1. *Let $M = (G, C, \ell)$ be a matrix formed by a set of genes, G, and a set of conditions, C. We say that a pair of non-empty sets (I, J) is a α–pattern, if $I \subseteq G$, $J \subseteq C$ and*

$$J = \{c \in C \mid \forall g, g' \in I, \mid \langle g, c\rangle - \langle g', c\rangle \mid \leq \alpha\}$$

The absolute value of the difference between two expression values of any pair of genes in I under the a specific condition from J is not greater than a threshold α.

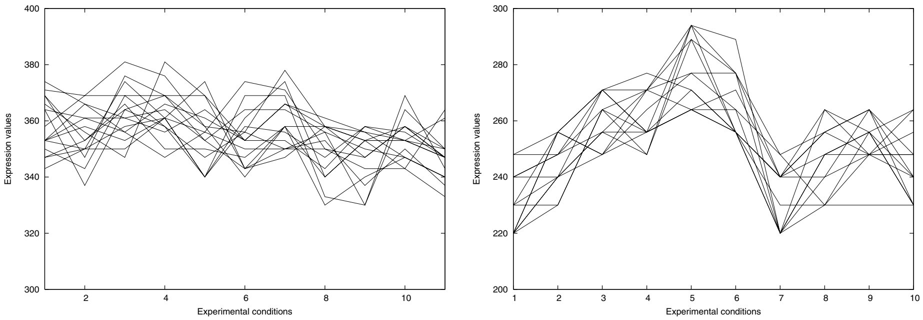

Fig. 1. An example of α–patterns. In both cases we can observe a band, that ranges from 0 up to α, in which the behaviors of the expression values of the genes under a subset of experimental conditions follow similar patterns. The x axis represents the experimental conditions. The y axis represents the gene expression values.

The threshold, α, determines a band in which the expression values of the genes evolve in a similar way across a set of experimental conditions. The co–expression of the genes in these types of bicluster can be observed in Figure 1 (biclusters obtained from Yeast dataset). The expression values in this behavior band can flow along a certain range without many changes, like in the bicluster on the left, or we can find biclusters with more variation in the expression values of the genes, as we can see in the bicluster on the right. In both cases, the shapes of the graphs show the quality of the α–patterns.

To compare the experimental results with those generated by Cheng & Church's algorithm, the *mean squared residue* (MSR) is used. Let (I, J) be a bicluster. The residue R of an element a_{ij} of the bicluster (I, J) is $R(a_{ij}) = a_{ij} - a_{iJ} - a_{Ij} + a_{IJ}$, where a_{iJ} is the mean of the ith row in the bicluster, a_{Ij} the mean of the jth column and a_{IJ} is the mean of all the elements within the bicluster. The mean squared residue, MSR, of (I, J) is defined as follows:

$$MSR(I, J) = \frac{1}{\mid I \mid\mid J \mid} \sum_{i \in I, j \in J} R^2(a_{ij})$$

This value is indicative of the coherence of values across both rows and columns. The lower the MSR, the stronger the coherence.

```
1 **Procedure:** $\alpha$PB
2 **Input:**
3   $M$ (data matrix $M = (G, C, \ell)$)
4   $\alpha$ (maximum difference between two expression values)
5   $\lambda$ (minimum length of the set of conditions for every bicluster)
6 **Output:**
7   $T$ (final set of biclusters)
8 **Method:**
9 Initialize $T = \emptyset$
11 **For** all bicluster $(I, J)$ with $\mid I \mid= 2$ and $\mid J \mid\geq \lambda$
12   inserted = false
13   **For** every bicluster $(I', J')$ in T
14     **If** $J' \subset J$ and Compatible($I$, $I'$, $J'$,$\alpha$)
15       add $I$ to $I'$
16     **If** $J \subseteq J'$ and Compatible($I$, $I'$, $J$,$\alpha$)
17       add $I'$ to $I$ and if $J == J'$ then inserted = true
18   **end For**
19   **If** not inserted
20     add $(I, J)$ to T
21 **end For**
22 **end** $\alpha$PB
```

Fig. 2. The α–Pattern Biclustering algorithm

3 Algorithm

Our approach, named αPB (α–Pattern Biclustering), is based on the definition of α–pattern provided above, in Def. 1. The aim is to obtain different biclusters with the maximum number of genes so that all of them have the next two properties:

- the difference between two expression values of any pair of these genes under the same condition in the bicluster is not greater than α.
- the number of conditions of each bicluster is not lower than λ.

The algorithm, illustrated in Figure 2, consists of two parts. In the first part all the valid α–patterns with only two genes are obtained, so that the algorithm analyzes all possible pairs of genes in the data matrix in order to find them, as we can see in line 11. The aim of the second part is to create new biclusters containing more than two genes. The biclusters will be stored in the set T, which is initialized as an empty set in line 9. We have designed a special tree data structure to implement the set T. In this tree, the nodes represent experimental conditions and leaves are sets of groups of genes that have a common group of conditions, i.e., to reach them the same path in the tree have to be followed. This structure is used to minimize the amount of memory used for storing the biclusters (they can be many thousands) and also to reduce the running time.

For each bicluster with two genes, (I,J), obtained in the first part, each bicluster stored in the set T, (I',J'), is analyzed in order to increase their number of genes (line 13). To do this, the groups of conditions and genes of these two biclusters have to be *compatible*.

Table 1. Example of α–patterns with two genes

$Bicluster$	$Genes$	$Conditions$	$Bicluster$	$Genes$	$Conditions$
$B1$	$\{1,2\}$	$\{2,3,4\}$	$B6$	$\{2,4\}$	$\{1,3,4\}$
$B2$	$\{1,3\}$	$\{1,2,3,4\}$	$B7$	$\{2,5\}$	$\{1,3,4\}$
$B3$	$\{1,4\}$	$\{1,3,4\}$	$B8$	$\{3,4\}$	$\{1,2,3,4\}$
$B4$	$\{1,5\}$	$\{1,3,4\}$	$B9$	$\{3,5\}$	$\{1,2,3,4\}$
$B5$	$\{2,3\}$	$\{2,3,4\}$	$B10$	$\{4,5\}$	$\{1,2,3\}$

Firstly, we check the compatibility between the groups of conditions of both biclusters, i.e., we check if these groups of conditions have one of the following properties (lines 14 and 16): J' $\subset$ J (J' is a subset of J), J $\subset$ J' (J is a subset of J') or J $=$ J' (J and J' are equal).

In the first case, the bicluster (I,J) could provide their genes to the bicluster in set T, (I',J') (line 15). On the contrary , in the last two cases, the set I' could be added to I (line 17). To deal with that increase of the number of genes we have to carry out a second verification. The group of genes of both biclusters, I and I', have to be compatible with the group of experimental conditions, which will be J', if we are in the first case in the preliminary checking, or J, in the last two cases. In the algorithm, the procedure *Compatible(I, I', J ,α)* is used to carry out this second verification (lines 14 and 16). Being I and I' groups of genes, J a group of conditions and α the maximum difference between two expressions values, the procedure *Compatible* returns "true" if:

$$\forall g \in I, g' \in I' \text{ and } c \in J, \mid \langle g, c\rangle - \langle g', c\rangle \mid \leq \alpha$$

which means that the absolute value of the difference between two expression values of any pair of genes of I and I', under the same condition of J, is not greater than a threshold α, for all the conditions in J.

Once all the biclusters on T have been analyzed, if the bicluster (I,J) did not add their genes to a bicluster on the tree structure with the same group of conditions, it will be add to the set T (lines 19 and 20).

For a better comprehension of our algorithm we present next a simple example. Consider that the biclusters with only two genes that have been obtained during the first part of the algorithm are those shown in Table 1. The first bicluster is: $B1 = \{\{g_1, g_2\}, \{c_2, c_3, c_4\}\}$. In the first iteration of the algorithm, the tree structure is empty, so $B1$ will be added to T without any modification: $T = \{(c_2, c_3, c_4) \rightarrow (g_1, g_2)\}$.

After that, the tree T has one branch, representing the group of conditions $\{c_2, c_3, c_4\}$ and one leave, representing the group of genes $\{g_1, g_2\}$. Next we are going to process the next bicluster: $B2 = \{\{g_1, g_3\}, \{c_1, c_2, c_3, c_4\}\}$. At this iteration, the tree T has one bicluster stored so we have to determine if the group of conditions and genes of both biclusters are compatible. In the case of the groups of conditions they are compatible because $\{c_2, c_3, c_4\} \subset \{c_1, c_2, c_3, c_4\}$, so in the next step we have to verify if B2 could add its genes to B1. To do this, we have to check if:

$$\forall g \in \{g_1, g_2\}, g' \in \{g_1, g_3\} \text{ and } c \in \{c_2, c_3, c_4\}, \mid \langle g, c\rangle - \langle g', c\rangle \mid \leq \alpha$$

As the two groups of genes have g_1 in common, we don't consider that gene. So we have to verify the previous property with the pair (g_2, g_3). As we can see in Table 1, it exists a bicluster formed by this pair of genes and with the same group of conditions as B1: $B5 = \{\{g_2, g_3\}, \{c_2, c_3, c_4\}\}$, so the property is verified and the genes of B2 can be added to B1 and T will be: $T = \{(c_2, c_3, c_4) \rightarrow (g_1, g_2, g_3)\}$.

Finally, B2 is also added to the tree structure because we didn't add its genes to a bicluster with the same group of conditions in T:

$$T = \{(c_2, c_3, c_4) \rightarrow (g_1, g_2, g_3), (c_1, c_2, c_3, c_4) \rightarrow (g_1, g_3)\}$$

At this point, the tree structure has two biclusters stored.

4 Method and Experimental Results

In this section, the method used to obtain biclusters using the algorithm αPB is described. The aim is to generate biclusters with the maximum number of genes and a low value of MSR. We have developed our experiments with a well known dataset: the *Saccharomyces Cerevisiae* cell cycle expression dataset. The *Yeast* dataset consists of a data matrix composed by 2884 genes (rows) and 17 experimental conditions (columns).

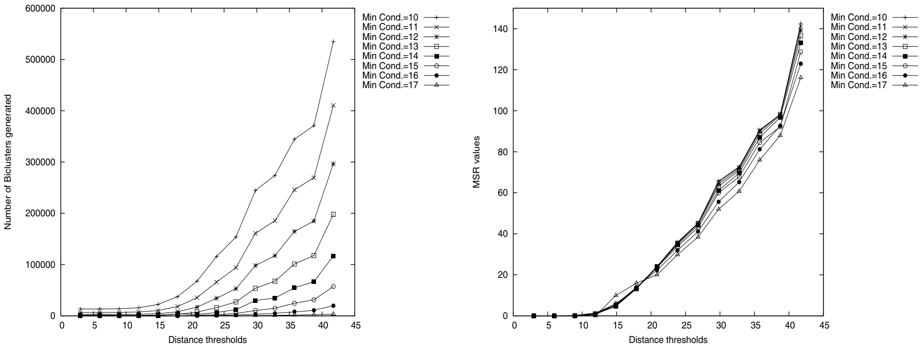

Fig. 3. Number of biclusters (left) and mean of the MSR value of the biclusters generated with different values of α (in X–axis) and λ. It can be observed that the MSR value and the number of biclusters generated increase when the parameters of the αPB algorithm are less restrictive.

The most important parameter of our algorithm is the distance threshold between expression values, α. The parameter α determines the level of similarity between gene behavior across the experimental conditions in the biclusters. To study the influence of the α parameter in the final results, an statistical study has been carried out using the Yeast dataset. We have run a special version of αPB with different values of α and λ parameters to collect information about the number and the mean of the MSR value of final biclusters. The conclusion obtained is that these two magnitudes increase when α and λ are less restrictive. That is, with a high value of the distance threshold and a low number of minimum experimental conditions allowed we can find more biclusters but

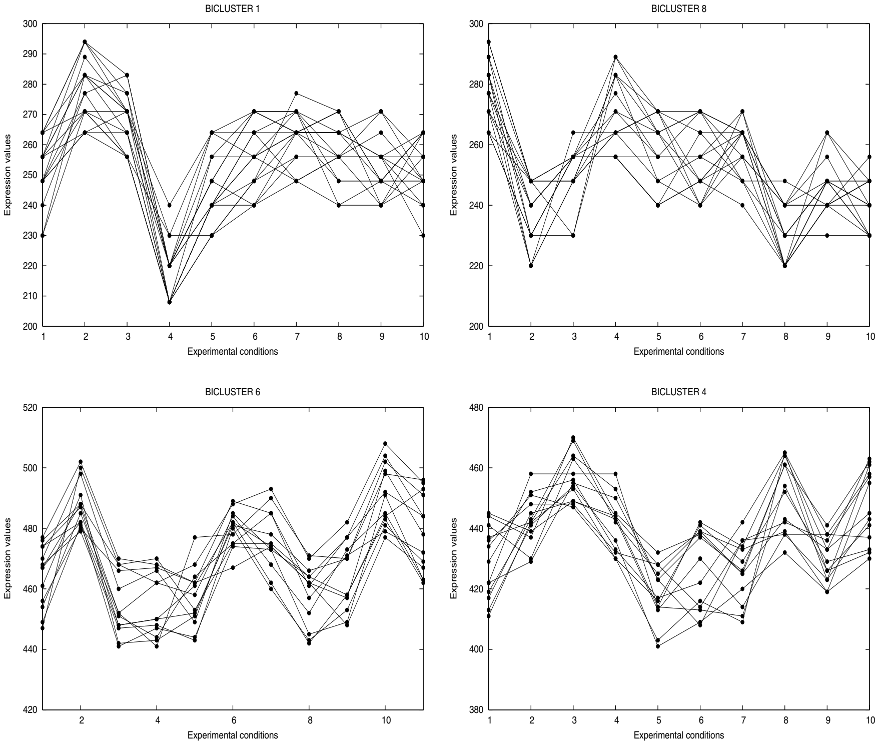

Fig. 4. An example of biclusters generated by αPB. The shapes of the graphs show the quality of biclusters. The y axis range is 100 in all cases.The x axis represents the experimental conditions. The y axis represents the gene expression values.

with less quality. Therefore, we have to achieve a trade–of between these magnitudes. Part of the results of this statistical study is showed in the graphs of Figure 3. In our experiment, we have chosen α=35 and, to make less restrictive the performance conditions, λ=10. With these values, as we can observe in the graphs of Figure 3, we obtain a low mean of MSR, between 80 and 90, and a great number of biclusters: 300000 approximately.

As a result, αPB uses 339421 biclusters with two genes to generates 322150 final biclusters. The maximum number of genes founded in a bicluster is 20, and the mean of the MSR value of all the biclusters is 85.11.

The criterion used to measure the quality of biclusters is the minimum mean squared residue, the maximum number of genes and the maximum number of conditions, in this order. Following these criteria, we have selected the best 100 biclusters among all having a number of genes between 13 and 20. The features of some of them are shown in Table 2. These biclusters present a small MSR value, i.e., it exists a great coherence across both genes and conditions. This similar behavior can be observed in Figure 4. These graphs show the evolution of the expression values of the set of genes under the set of conditions in a behavior band that ranges from 0 up to α. We obtain biclusters

Table 2. Information about biclusters generated by αPB

Bicluster	Genes	Conditions	MSR
1	17	10	73.09
2	15	10	73.48
3	16	11	66.16
4	13	10	72.02
5	20	10	104.95
6	13	11	65.19
7	14	10	71.03
8	16	10	87.41
9	15	10	75.32

Table 3. Performance comparison between CC and αPB

Alg.	MSR	Vol.	Mean Genes	Mean Cond.
CC	204.29	1576,98	166.71	12.09
SEBI	202.68	204.67	13.20	15.44
αPB	81.88	145,27	14.37	10.11

with high number of genes as well, being 20 the maximum value of genes in a bicluster. It is specially interesting that the range of values is very small regarding the range of expression levels (from 0 to 600).

In Table 3, we compare our 100 best biclusters and their average values with those obtained by the algorithms of Cheng & Church (CC) and Aguilar & Divina (SEBI). As we can observe, αPB obtains better results with regard to the MSR value. The averaged volume, i.e., the number of genes multiplies by the number of conditions, and the average of conditions in biclusters are lower. The average of genes is lower than the same measure obtained by CC but greater if it is compared with SEBI. The most interesting property of α–patterns found by αPB is that it provides biclusters with very low mean squared residue in comparison to the CC and SEBI algorithms, while maintaining a good number of genes, between 13 and 20.

5 Conclusions

In this work we present a new technique to discover a certain type of biclusters in gene expression data. These biclusters, named α–patterns, are based on the distance between the expression values of genes. The distance threshold α determines a band in which the expression values of a subset of genes have similar behavior under a subset of conditions. Our approach, named αPB, provides a group of different biclusters with highly–related genes and very low mean squared residue. Results show interesting biclusters in comparison to Cheng & Church approach (CC, based on a greedy strategy) and Aguilar & Divina's ones (SEBI, based on an evolutionary technique) . Our approach obtains biclusters with much less number of genes than CC and the lowest mean squared residue.

References

1. Dougherty, E., Barrera, J.: Inference from Clustering with Application to Gene-Expression Microarrays. Journal of Computational Biology 9(1), 105–126 (2002)
2. Madeira, S., Oliveira, A.: Biclustering Algorithms for Biological Data Analysis: A Survey. IEEE/ACM Transactions on Computational Biology and Bioinformatics 1(1), 24–45 (2004)
3. Hartigan, J.A.: Direct clustering of a data matrix. Journal of the American Statistical Association 67(337), 123–129 (1972)
4. Cheng, Y., Church, G.M.: Biclustering of expression data. In: Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology, pp. 93–103 (2000)
5. Yang, J., Wang, W.: Improving Performance of Bicluster Discovery in a Large Data Set. In: 6th ACM International Conference on Research in Computational Molecular Biology, Poster (2002)
6. Yang, J., Wang, W.: Enhanced biclustering on expression data. In: 3rd IEEE Conference on Bioinformatics and Bioengineering, pp. 321–327 (2003)
7. Lazzeroni, L.: Owen: Plaid models for gene expression data. Technical Report Stanford University (2000)
8. Shamir, R., Sharan, R.: Discovering statistically significant biclusters in gene expression data. Bioinformatics 19(Suppl. 1), 136–144 (2002)
9. Liu, J., Yang, J.: Biclustering in Gene Expression Data by Tendency. In: IEEE Computational Systems Bioinformatics Conference, pp. 183–193 (2004)
10. Ben-Dor, A., Chor, B.: Discovering local structure in gene expression data: The Order Preserving Submatrix Problem. In: 6th ACM International Conference on Research in Computational Molecular Biology (2002)
11. Gerstein, M., Chang, J.: Spectral Biclustering of Microarray Data: Coclustering Genes and Conditions. Journal Genome Research 13(4), 703–716 (2003)
12. Aguilar, J.S., Divina, F.: Evolutionary Biclustering of Microarray Data. In: 3rd European Workshop on Evolutionary Bioinformatics (2005)