Iterated Local Search for Biclustering of Microarray Data

Wassim Ayadi[1,2], Mourad Elloumi[2], and Jin-Kao Hao[1]

[1] LERIA, University of Angers, 2 Boulevard Lavoisier, 49045 Angers, France
[2] UTIC, Higher School of Sciences and Technologies of Tunis, 1008 Tunis, Tunisia
{ayadi, hao}@info.univ-angers.fr; mourad.elloumi@fsegt.rnu.tn

Abstract. In the context of microarray data analysis, biclustering aims to identify simultaneously a group of genes that are highly correlated across a group of experimental conditions. This paper presents a Biclustering Iterative Local Search (BILS) algorithm to the problem of biclustering of microarray data. The proposed algorithm is highlighted by the use of some original features including a new evaluation function, a dedicated neighborhood relation and a tailored perturbation strategy. The BILS algorithm is assessed on the well-known yeast cell-cycle dataset and compared with two most popular algorithms.

Key words: Analysis of DNA microarray data, biclustering, evaluation function, iterative local search.

1 Introduction

With the fast advances of DNA Microarray technologies, more and more gene expression data are made available for analysis. In this context, *biclustering* has been recognized as a remarkably effective method for discovering several groups of subset of genes associated with a subset of conditions. These groups are called *biclusters*. Biclusters can be used for various purposes, for instance, they are useful to discover genetic knowledge, such as gene annotation or gene interaction, and to understand various genetic diseases.

Formally, DNA microarray data is usually represented by a data matrix $M(I, J)$, where the i^{th} row, $i \in I=\{1, 2, \ldots, n\}$, represents the i^{th} gene, the k^{th} column, $k \in J=\{1, 2, \ldots, m\}$, represents the k^{th} condition and the cell $M[i, k]$ represents the expression level of the i^{th} gene under the k^{th} condition. A *bicluster* of M is a couple (I', J') such that $I' \subseteq I$ and $J' \subseteq J$.

The *biclustering problem* consists in extracting from a data matrix $M(I, J)$ a group of biclusters that maximize a given evaluation function. The biclustering problem is known to be NP-hard [10, 22]. In the literature there are two main approaches for biclustering: the *systematic search* approach and the *stochastic search* or *metaheuristic* approach. Notice that most of these approaches are approximate methods.

The *systematic search* approach includes greedy algorithms [6, 9, 10, 29], divide-and-conquer algorithms [17, 26] and enumeration algorithms [4, 20]. The *meta-heuristic* approach includes neighbourhood-based algorithms [8], GRASP [12, 13] and evolutionary algorithms [15, 16, 23]. A recent review of various biclustering algorithms for biological data analysis is provided in [3].

In this paper, we present a first adaptation of Iterative Local Search (ILS) to the biclustering problem. The resulting algorithm, called BILS, integrates several original features. BILS employs a new evaluation function for the assessment of biclusters. In BILS, we introduce a dedicated neighborhood relation which allows the search to improve gradually the quality of biculters. To allow the search to escape from local optima, BILS uses a randomized, yet guided perturbation strategy.

To assess the performance of BILS, we applied BILS to the well-known yeast cell-cycle dataset and validated the extracted biclusters using external biological information by determining the functionality of the genes of the biclusters from the Gene Ontology database [2] using GOTermFinder tool[3]. Genes belonging to our biclusters were found to be significantly enriched with GO terms with very small p-values. We also use the web tool FuncAssociate [7] to compute the adjusted p-values. Our biclusters were found to be statistically significant with adjusted p-values < 0.001. We also compared our algorithm with two popular biclustering algorithms of Cheng and Church (CC) [10] and OPSM [6].

The remainder of the paper is organized as follows: In section 2, we describe our new biclustering algorithm. In section 3, we carry out an experimental study of BILS and assess its results using the above cited web-tools. Finally, in the last section, we present our conclusion and perspective.

2 The BILS algorithm

2.1 Iterated local search

Iterated Local Search can be described by a simple computing schema [19]. A fundamental principle of ILS is to exploit the tradeoff between intensification and diversification. Intensification focuses on optimizing the objective function as far as possible within a limited search region while diversification aims to drive the search to explore new promising regions of the search space. The diversification mechanism of ILS–perturbation operator–has two aims: one is to jump out of the local optimum trap; the other is to lead the search procedure to a new promising region.

From the operational point of view, An ILS algorithm starts with an initial solution and performs local search until a local optimum is found. Then, the current local optimum solution is perturbed and another round of local search is performed with the perturbed solution.

Our BILS algorithm follows this general ILS schema. It uses a Hill-climbing (HC) algorithm as its local search procedure. In the rest of this section, we explain the main ingredients of this HC algorithm as well as the perturbation-based diversification strategy.

[3] http://db.yeastgenome.org/cgi-bin/GO/goTermFinder

2.2 Preprocessing step: construction of the Behavior Matrix

Prior to the search step using ILS, our method first uses a preprocessing step to transform the input data matrix M to a *Behavior Matrix* M'. This preprocessing step aims to highlight the trajectory patterns of genes. Indeed, according to [21, 24, 27], in microarray data analysis, genes are considered to be in the same cluster if their trajectory patterns of expression levels are similar across a set of conditions. In our case, each column of M' represents the trajectory of genes between a pair of conditions in the data matrix M. The whole M' matrix provides useful information for the identification of related biclusters and the definition of a meaningful neighborhood and perturbation strategy.

Formally, the Behavior Matrix M' is constructed progressively by merging a pair of columns (conditions) from the input data matrix M. Since M has n rows and m columns, there is $m(m-1)/2$ distinct combinations between columns, represented by J''. So, M' has n rows and $m(m-1)/2$ columns. M' is defined as follows:

$$M'[i,l] = \begin{cases} 1 & \text{if } M[i,k] < M[i,q] \\ -1 & \text{if } M[i,k] > M[i,q] \\ 0 & \text{if } M[i,k] = M[i,q] \end{cases} \tag{1}$$

with $i \in [1..n]$, $l \in [1..J'']$, $k \in [1..m-1]$, $q \in [1..m]$ and $q > k+1$.

Using M', we can observe the behavior of each gene through all the combined conditions. In our case, the combination of all conditions gives useful information since a bicluster may contains a subset of non contiguous conditions.

2.3 Initial solutions and basic search process

Given the Behavior Matrix M', our BILS algorithm explores iteratively different biclusters. To do this, BILS needs an initial bicluster (call it s_0) as its starting point. This initial bicluster can be provided by any means. For instance, this can be done randomly with a risk of starting with an initial solution of bad quality. A more interesting strategy is to employ a fast greedy algorithm to obtain rapidly a bicluster of reasonable quality. We use this strategy in this work and adopt two well-known algorithms: one is presented by Cheng and Church [10] and the other is called OPSM which is introduced in [6].

Starting from this initial solution, BILS will try to find iteratively biclusters of better and better quality. Basically, the improvement is realized by removing a "bad" genes from the current bicluster and adding one or more other "better" genes. Each application of this dual drop/add operation generates a new bicluster from the current bicluster. The way of identifying the possible genes to drop and to add defines the so-called neighborhood which is explained in detail in section 2.6.

2.4 Solution representation and search space

A candidate solution is simply a bicluster and represented by $s = (I', J')$. As explained in the next section, our algorithm explores different biclusters with variable number of genes and a fixed number of conditions. The search space is thus determined by the number k of genes in the initial bicluster and has size of 2^g where $g = n - k$.

2.5 Evaluation function

For a given solution (bicluster), its quality is assessed by an evaluation function. One of the most popular evaluation functions in the literature is called *Mean Squared Residue* (MSR) [10]. MSR has been used by several biclustering algorithms [9,13,23]. Yet MSR is known to be deficient to assess correctly the quality of certain types of biclusters like multiplicative models [1,25,29,9]. Recently, Teng and Chan [29] proposed another function for bicluster evaluation called Average Correlation Value (ACV). However, the performance of ACV is known to be sensitive to errors [9]. Both MSR and ACV are designed to be applied to the initial data matrix M. In our case, since M is preprocessed to obtain M', the above mentioned evaluation functions cannot be applied. For these reasons, we propose a new evaluation function $\mathbb{S}$ to evaluate a bicluster.

Given a candidate solution (a bicluster) $s = (I', J')$, the quality of s is assessed *via* the following score function $\mathbb{S}(s)$:

$$\mathbb{S}(s) = \frac{\sum\limits_{i \in I'} \sum\limits_{j \in I', j > i+1} \mathcal{F}_{ij}(g_i, g_j)}{|I'|(|I'| - 1)/2} \tag{2}$$

with $\mathcal{F}_{ij}(.,.)$ being defined by:

$$\mathcal{F}_{ij}(g_i, g_j) = \frac{\sum\limits_{l \in J''_{s_0}} T(M'[i,l] = M'[j,l])}{|J''_{s_0}|} \tag{3}$$

where

- $T(Func)$ is true, if and only if $Func$ is true, and $T(Func)$ is false otherwise.
- $i \in I'$, $j \in I'$ and $i \neq j$, when $\mathcal{F}$ is used by $\mathbb{S}$ and, $i \in I$, $j \in I$ and $i \neq j$ otherwise.
- $|J''_{s_0}|$ is the cardinality of the subset of conditions in M' obtained from s_0,
- $0 \leq \mathcal{F}_{ij}(g_i, g_j) \leq 1$.

In fact, each $\mathcal{F}$ score assesses the quality of a pair of genes (g_i, g_j) under the subset of conditions of s. A high (resp. low) $\mathcal{F}_{ij}(g_i, g_j)$ value, *close* to 1 (resp. *close* to 0), indicates that the genes (g_i, g_j) (under the given conditions) are strongly (resp. weakly) correlated.

Given two pairs of genes (g_i, g_j) and (g'_i, g'_j), it is then possible to compare them: (g_i, g_j) is better than (g'_i, g'_j), when $\mathcal{F}_{ij}(g_i, g_j) > \mathcal{F}_{ij}(g'_i, g'_j)$.

Furthermore, $\mathbb{S}(s)$ is an average of $\mathcal{F}_{ij}(g_i, g_j)$ for each pair of genes in s. So, $0 \leq \mathbb{S}(s) \leq 1$. As $\mathcal{F}_{ij}(g_i, g_j)$, a high (resp. low) $\mathbb{S}(s)$ value, *close* to 1 (resp. *close* to 0), indicates that the solution s is strongly (resp. weakly) correlated.

Now given two candidate solutions s and s', s is better than s' if $\mathbb{S}(s) > \mathbb{S}(s')$.

2.6 Move and neighborhood

One of the most important features of a local search algorithm is its neighborhood. In a local search algorithm, applying a move operator mv to a candidate solution s leads to a new solution s', denoted by $s' = s \oplus mv$. Let $\Gamma(s)$ be the set of all possible moves which can be applied to s, then the neighborhood $N(s)$ of s is defined by: $N(s) = \{s \oplus mv | mv \in \Gamma(s)\}$.

In our case, the move is based on the drop/add operation which removes a gene $\{g_i | i \in I'\}$ from the solution s and add another gene $\{g_v | v \notin I'\}$ or several other genes $\{g_v, \ldots, g_w | v \notin I', \ldots, w \notin I'\}$ to s.

The move operator can be defined as follows. Let $s = (I', J')$ be a solution and let $\lambda \in [0..1]$ be a fixed quality *threshold* (See Section 2.5 for quality evaluation). For each $i \in I', j \in I', r \in I'$ and $i \neq j \neq r$, we first choose a pair of genes (g_i, g_j) such that $\mathcal{F}_{ij}(g_i, g_j) < \lambda$. Such a pair of genes shows that they contributes negatively to the quality of the bicluster when they are associated. Now we look for another pair of genes (g_j, g_r) satisfying $\mathcal{F}_{jr}(g_j, g_r) \geq \lambda$. By this choice, we know that g_j contributes positively to the quality of the bicluster when it is associated with g_r. Notice that for both choices, ties are broken at random in order to introduce some diversification in the move operator.

Finally, we remove g_i which is a bad gene among the genes belonging to I' and we add all the genes $\{g_v, \ldots, g_w | v \notin I', \ldots, w \notin I'\}$ such that the values $\mathcal{F}_{rv}(g_r, g_v), \ldots, \mathcal{F}_{rw}(g_r, g_w)$ are higher than or equal to λ. Such an operator clearly help improve the quality of a bicluster, but also maximize the bicluster size [14, 23].

Applying the move operator to a solution s leads to a new bicluster s', called neighboring solution or simply neighbor. For a given bicluster s, all possible neighbors define its neighborhood $N(s)$. It is clear that a neighboring solution s' has at least as many genes as in the original solution s.

2.7 The general BILS procedure

The general BILS procedure is given in Algorithm 1. Starting from an initial solution (call it current solution s, see section 2.3), our BILS algorithm uses the Hill-climbing strategy to explore the above neighborhood. At each iteration, we move to an improving neighboring solution $s' \in N(s)$ according to the evaluation function $\mathbb{S}(s)$. This Hill-climbing based intensification phase stops when no improving neighbor can be found in the neighborhood. So, the last solution is the best solution found and corresponds to a local optimum. At this point, BILS triggers a diversification phase by perturbing the best solution to generate a new starting point for the next round of the search.

Our perturbation operator changes the best local optimum by deleting randomly 10% of genes of the best solution and adding 10% of genes among the best genes that are not included in the best solution. This perturbed solution is used by BILS as its new starting point.

The whole BILS algorithms stops when the best bicluster reaches a fixed quality or when the best solution found is not updated for a fixed number of perturbations.

Algorithm 1 General BILS Procedure

```
1: Input: An initial bicluster $s_0$, quality threshold $\lambda$
2: Output: The best bicluster
3: Create the Behaviour Matrix $M'$
4: Compute $\mathcal{F}$ for all pairs of genes to create $\Gamma(s_0)$
5: $s = s_0$ // current solution
6: repeat
7:    repeat
8:       Choose a pair of genes $(g_i, g_j)$ belonging to $s$ such that $\mathcal{F}_{ij}(g_i, g_j) < \lambda$
9:       Choose a pair of genes $(g_j, g_r)$ belonging to $s$ such that $\mathcal{F}_{jr}(g_j, g_r) \geq \lambda$
10:      Identify all genes $g_v$, $v \notin I'$ such that $\mathcal{F}_{rv}(g_r, g_v) \geq \lambda$
11:      Generate neighbor $s'$ by dropping $g_i$ from $s$ and adding all $g_v$
12:      if $(\mathbb{S}(s') \geq \mathbb{S}(s))$ then $s = s'$
13:      endif
14:   until (no improving neighbor can be found in $N(s)$)
15:   Generate a new solution $s$ by perturbing randomly 10% of the best solution
16: until (stop condition is verified)
17: Return $s$
```

3 Experimental Results

3.1 Dataset and experimental protocol

In order to analyze the effectiveness of the proposed algorithm, we used the well-known yeast cell-cycle microarray dataset. The yeast cell-cycle dataset is described in [28]. It is processed in [10] and publicly available from [11]. It contains the expression profiles of more than 6000 yeast genes measured at 17 conditions over two complete cell cycles. In our experiments we use 2884 genes selected by [10].

The obtained results have been compared with two popular biclustering algorithms: the one proposed by Cheng and Church (CC) [10] and OPSM described in [6]. For these reference algorithms, we have used *Biclustering Analysis Toolbox* (BicAT) which is a recent software platform for clustering-based data analysis that integrates these biclustering algorithms [5].

For this experiment, the λ threshold of BILS is experimentally set to 0.7. In fact, for each experiment ten values are tested between 0.1 and 1 with a stepwise of 0.1. With $\lambda = 0.7$, we have obtained the lowest p-values. The threshold δ of CC is selected as 300 like used in [10] and the default parameter setting is used for OPSM. With these algorithms, we have obtained 10 biclusters for CC and 14 biclusters for OPSM. Post-filtering was applied in order to eliminate insignificant biclusters like Cheng *et al.* [9]. This led to 8 biclusters CC and for 10 biclusters for OPSM. These biclusters are used as initial solutions for BILS and we compare the outputs of BILS with these initial biclusters.

The two web tools Funcassociate [7] and GoTermFinder[4] are used to evaluate statistically and biologically the biclusters.

[4] http://db.yeastgenome.org/cgi-bin/GO/goTermFinder

Our algorithm is run on a PC with 3.00GHz CPU and 3.25Gb RAM. Computing time is not reported, but let us mention that to improve one bicluster it takes between 3 and 11 minutes.

3.2 Statistical and biological significance evaluation

Statistical significance of the biclusters is obtained by using the Funcassociate [7] web tool to compute the p-values and the adjusted p-values.

First, we asses the quality of the group of 18 biclusters obtained by BILS when it is applied to the 8 initial biclusters provided by CC and 10 initial biclusters given by OPSM. Funcassociate is used to compute the adjusted p-values of each of our 18 biclusters, leading always to an adjusted p-values < 0.001. This indicates that all these biclusters are statistically significant.

Now we turn our attention to the interpretation of results using the p-values. In fact, the p-values show how well they match with the known gene annotation. The closer the p-value is to zero, the more significant is the association of the particular Gene Ontology (GO) with the group of genes. For this purpose, we decide to examine for each algorithm only two biclusters: the bicluster having the maximum p-value and the one having the minimum p-value. Let B_xx_{MaxP} (resp. B_xx_{MinP}) denote these biclusters for algorithm $xx =$ CC or $xx =$ OPSM.

Table 1 summarizes the largest (column 2) and the smallest (column 3) p-values of the eight biclusters obtained from CC and the ten biclusters obtained from OPSM. The obtained biclusters from these algorithms with largest/smallest p-values are improved with BILS (row 3 for CC and 5 for OPSM). For instance, the element 0.000010 at row 2 and column 2 is the p-value of the bicluster B_CC_{MaxP} of CC while the element 2.220e-17 at row 3 and column 2 is the p-value of the improved bicluster B_CC_{MaxP} by BILS.

From the table, we see that BILS successfully improves the biclusters of CC and OPSM. In fact, both the maximum and minimum p-values of BILS are always better than those of CC and OPSM. This demonstrates that BILS is able to replace bad genes of the candidate solution by good genes by applying our move operator. Thus we can say that the biclusters of BILS are more statistically significant than those of CC and OPSM.

Algorithms	Maximum p-value	Minimum p-value
CC	0.000010	4.096e-40
BILS	2.220e-17	2.860e-70
OPSM	0.0000012	1.587e-13
BILS	1.156e-10	4.865e-24

Table 1. P-values of the genes of the biclusters for BILS, CC and OPSM.

In addition to the above statistical significance validation, we also apply the GoTermFinder web tool on the biclusters used at the Table 1 to evaluate their biological significance, i.e., to show significant enrichment with respect to a specific GO annotation, in terms of associated biological processes, molecular functions and cellular components respectively compared to CC and OPSM.

For this, Table 2 and 3 describe the top GO terms of the three categories with the lowest p-values. The value within parentheses after each GO term, e.g., Table 2 second column third line, such as (4.54e-05) indicates the statistical significance which is provided by the p-value. We observe that BILS can obtain improved biclusters not only in terms of p-values, i.e., quality of biclusters, but also in terms of GO annotation. For example Table 2 (resp. Table 3) shows that CC (resp. OPSM) can not identify any biological process and molecular functions (resp. biological process and cellular component) for the bicluster B_CC_{MaxP} (resp. B_OPSM_{MinP}). However, BILS can produce biclusters with all categories, i.e., biological processes, molecular functions and cellular components. This shows that our algorithm is able to identify biological significant biclusters.

Algorithms	Biological Process	Molecular function	Cellular component
CC (B_CC_{MaxP})	unknown	unknown	Cytoplasm (0.00932)
BILS$_{CC}$: improved B_CC_{MaxP} by BILS	Maturation of SSU-rRNA (4.54e-05) Maturation of SSU-rRNA from tricistronic rRNA transcript(SSU-rRNA, 5.8S rRNA, LSU-rRNA) (0.00088) Cell cycle (0.00107)	structural constituent of ribosome (4.14e-17) Structural molecule activity (1.97e-15)	cytosolic ribosome (2.94e-21) ribosomal subunit (4.27e-17) cytosolic part (2.04e-16)
CC (B_CC_{MinP})	translation (8.33e-23) cellular protein metabolic process (3.17e-10) gene expression (6.48e-10)	structural constituent of ribosome (1.03e-36) structural molecule activity (3.91e-28) helicase activity (0.00021)	cytosolic ribosome (7.83e-42) ribosome (3.80e-36) cytosolic part (1.82e-35)
BILS$_{CC}$: improved B_CC_{MinP} by BILS	translation (2.86e-35) cellular protein metabolic process (2.59e-16) cellular macromolecule biosynthetic process (1.74e-15)	structural constituent of ribosome (2.50e-70) Structural molecule activity (6.06e-54) translation factor activity, nucleic acid binding (0.00445)	cytosolic ribosome (1.05e-76) ribosomal subunit (1.08e-68) cytosolic part (1.01e-66)

Table 2. Most significant shared GO terms (biological process, molecular function, cellular component) of CC and BILS for two biclusters on yeast cell-cycle dataset.

Algorithms	Biological Process	Molecular function	Cellular component
OPSM (B_OPSM_{MaxP})	sister chromatid segregation (0.00337) chromosome segregation (0.00478) microtubule-based process (0.00588)	unknown	spindle (0.00196) microtubule cytoskeleton (0.00295) chromosomal part (0.00991)
BILS$_{OPSM}$: improved B_OPSM_{MaxP} by BILS	cellular component organization (1.71e-07) nucleic acid metabolic process (1.72e-06) cellular nitrogen compound metabolic process (7.88e-06)	structural constituent of cytoskeleton (0.00099) RNA polymerase II transcription factor (0.00640)	nucleus (3.83e-12) nuclear part (3.91e-09) chromosomal (2.26e-08)
OPSM (B_OPSM_{MinP})	unknown	oxidoreductase activity (6.78e-06) oxidoreductase activity, acting on CH-OH group of donors (0.00075) oxidoreductase activity, acting on peroxidase as acceptor (0.00078)	unknown
BILS$_{OPSM}$: improved B_OPSM_{MinP} by BILS	response to stimulus (0.00092) response to stress (0.00454)	structural constituent of ribosome (9.19e-24) structural molecule activity (3.78e-12) oxidoreductase activity (2.36e-05)	cytosolic ribosome (1.09e-23) ribosomal subunit (3.28e-23) cytosolic part (7.35e-22)

Table 3. Most significant shared GO terms (biological process, molecular function, cellular component) of OPSM and BILS for two biclusters on yeast cell-cycle dataset.

4 Conclusion and future work

In this paper, we have presented a new biclustering algorithm using Iterative Local Search (BILS). BILS combines a dedicated Hill-climbing based local search procedure and a perturbation strategy. For the intensification purpose, BILS employs a new evaluation function and a dedicated neighborhood relation. We have tested and assessed our algorithm on the yeast cell-cycle dataset. The experimental results show that the BILS algorithm can successfully improve all biclusters of CC and OPSM according to statistical and biological evaluation criteria.

The work reported in this paper correspond in fact to an ongoing study. Several improvements to the proposed work can be envisaged. One immediate possibility would be to study alternative neighborhoods to introduce more biological knowledge to provide more effective guidance of the local search process. Another natural extension would be to reinforce the basic local search procedure by more powerful metaheuristics such as Tabu Search. Moreover, BILS explores the space of biclusters by changing only the subset of genes of a bicluster without changing the conditions of the initial bicluster. It is natural to design similar strategies to optimize the subset of conditions of a bicluster or eventually to optimize simultaneously both the set of genes and conditions. Finally, another possible experimentation is to assess the algorithm on a synthetic data.

5 Acknowledgements

We are grateful for comments by the referees that have improved the paper. This research was partially supported by the region Pays de la Loire (France) with its 'Bioinformatics Program' (2007-2010).

References

1. J. S. Aguilar-Ruiz. Shifting and scaling patterns from gene expression data. *Bioinformatics*, 21:3840–3845, 2005.
2. M. Ashburner, C.A. Ball, J.A. Blake, D. Bolstein, H. Butler, M. Cherry, A.P. Davis, K. Dolinski, S.S. Dwight, J.T. Eppig, M.A. Harris, D.P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J.C. Matese, J.E. Richardson, M. Ringwald, G.M. Rubinand, and G. Sherlock. Gene ontology: tool for the unification of biology. the gene ontology consortium. *Nature Genetics, 25*, pages 25–29, 2000.
3. W. Ayadi and M. Elloumi. *Algorithms in Computational Molecular Biology: Techniques, Approaches and Applications*, chapter Biclustering of Microarray Data. John Wiley & Sons Inc, 2010, to appear.
4. W. Ayadi, M. Elloumi, and J. K. Hao. A biclustering algorithm based on a bicluster enumeration tree : Application to dna microarray data. *BioData Mining*, 2:9, 2009.
5. S. Barkow, S. Bleuler, A. Prelic, P. Zimmermann, and E. Zitzler. Bicat: a biclustering analysis toolbox. *Bioinformatics*, 22(10):1282–1283, 2006.

6. A. Ben-Dor, B. Chor, R. Karp, and Z. Yakhini. Discovering local structure in gene expression data: the order-preserving submatrix problem. In *RECOMB '02: Proceedings of the sixth annual international conference on Computational biology*, pages 49–57, New York, NY, USA, 2002. ACM.
7. G.F. Berriz, J.E. Beaver, C. Cenik, M. Tasan, and F.P. Roth. Next generation software for functional trend analysis. *Bioinformatics*, 25(22):3043–3044, 2009.
8. K. Bryan, P. Cunningham, and N. Bolshakova. Application of simulated annealing to the biclustering of gene expression data. In *IEEE Transactions on Information Technology on Biomedicine, 10(3)*, pages 519–525, 2006.
9. K.O. Cheng, N.F. Law, W.C. Siu, and A.W. Liew. Identification of coherent patterns in gene expression data using an efficient biclustering algorithm and parallel coordinate visualization. *BMC Bioinformatics*, 9(210), 2008.
10. Y. Cheng and G. M. Church. Biclustering of expression data. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 93–103. AAAI Press, 2000.
11. Y. Cheng and G.M. Church. Biclustering of expression data. Technical report, (supplementary information), 2006.
12. S. Das and S.M. Idicula. Application of reactive grasp to the biclustering of gene expression data. In *ISB '10: Proceedings of the International Symposium on Biocomputing*, pages 1–8, New York, NY, USA, 2010. ACM.
13. A. Dharan and A.S. Nair. Biclustering of gene expression data using reactive greedy randomized adaptive search procedure. *BMC Bioinformatics*, 10 (Suppl 1):S27, 2009.
14. F. Divina and J.S. Aguilar-Ruiz. Biclustering of expression data with evolutionary computation. In *IEEE Transactions on Knowledge and Data Engineering*, pages 590–602, Vol. 18, No. 5 May 2006.
15. F. Divina and J.S. Aguilar-Ruiz. A multi-objective approach to discover biclusters in microarray data. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 385–392, New York, NY, USA, 2007. ACM.
16. C.A. Gallo, J.A. Carballido, and I. Ponzoni. Microarray biclustering: A novel memetic approach based on the pisa platform. In *EvoBIO '09: Proceedings of the 7th European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, pages 44–55, Berlin, Heidelberg, 2009. Springer-Verlag.
17. J. A. Hartigan. Direct clustering of a data matrix. *Journal of the American Statistical Association*, 67(337):123–129, 1972.
18. H. Hoos and T. Stutzle. *Stochastic Local Search: Foundations and Applications.* Morgan Kaufmann Publishers Inc. San Francisco, CA, USA, 2004.
19. H.R. Lourenco, O. Martin, T. Stutzle. Iterated local search. In: Glover, F., Kochenberger, G. (eds.) Handbook of Meta-heuristics, pp. 321353. Springer, Heidelberg 2003.
20. J. Liu and W. Wang. Op-cluster: Clustering by tendency in high dimensional space. *IEEE International Conference on Data Mining*, pages 187 – 194, 2003.
21. Y. Luan and H. Li. Clustering of time-course gene expression data using a mixed-effects model with B-splines. *Bioinformatics*, 19(4):474–482, 2003.
22. Sara C. Madeira and Arlindo L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 1(1):24–45, 2004.
23. S. Mitra and H. Banka. Multi-objective evolutionary biclustering of gene expression data. *Pattern Recogn.*, 39(12):2464–2477, 2006.

24. S.D. Peddada and E.K. Lobenhofer and L. Li and C.A. Afshari and C.R. Weinberg and D.M. Umbach. Gene selection and clustering for time-course and dose-response microarray experiments using order-restricted inference. *Bioinformatics*, 19(7):834–841, 2003.
25. B. Pontes, F. Divina, R. Giráldez, and J.S. Aguilar-Ruiz. Virtual error: A new measure for evolutionary biclustering. In *Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, pages 217–226, 2007.
26. A. Prelic, S. Bleuler, P. Zimmermann, P. Buhlmann, W. Gruissem, L. Hennig, L. Thiele, and E. Zitzler. A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics*, 22(9):1122–1129, 2006.
27. A. Schliep and A. Schonhuth and C. Steinhoff. Using hidden Markov models to analyze gene expression time course data. *Bioinformatics*, 19 Suppl 1:i255–i263, 2003.
28. S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church. Systematic determination of genetic network architecture. *Nature Genetics*, 22:281–285, 1999.
29. Li Teng and Laiwan Chan. Discovering biclusters by iteratively sorting with weighted correlation coefficient in gene expression data. *J. Signal Process. Syst.*, 50(3):267–280, 2008.