Soft Computing
https://doi.org/10.1007/s00500-018-3394-4

METHODOLOGIES AND APPLICATION

Evolutionary biclustering algorithms: an experimental study on microarray data

Ons Maâtouk[1,2] · Wassim Ayadi[2,3] · Hend Bouziri[1] · Béatrice Duval[2]

Abstract
The extraction of knowledge from large biological data is among the main challenges of bioinformatics. Several data mining techniques have been proposed to extract data; in this work, we focus on biclustering which has grown considerably in recent years. Biclustering aims to extract a set of genes with similar behavior under a condition set. In this paper, we propose an evolutionary biclustering algorithm and we analyze its performance by varying its genetic components. Hence, several versions of the evolutionary biclustering algorithm are introduced. Further, an experimental study is achieved on two real microarray datasets and the results are compared to other state-of-the-art biclustering algorithms. This thorough study allows to retain the best combination of operators among the various experienced choices.

Keywords Biclustering · Evolutionary algorithm · Genetic operators · Microarray data · Data mining

1 Introduction

In recent years, biological research has grown considerably. Indeed, before the 1980s, it was enough to cross different species to make great biological and genetic progress. However, nowadays, the researchers resort to more complex techniques. These techniques require the processing of very large masses of information that cannot be addressed without resorting to computer sciences. The necessity of using computational methods has generated a new discipline: bioinformatics. It plays an increasingly important role in the study of modern biology. It evolves according to new problems of molecular biology. Several definitions have been proposed. However, the majority of these definitions consider bioinformatics as an interaction between biology, information technology and computer science.

Among the fundamental objectives of this discipline, there is the integration of data of different natures. Indeed, bioinformatics is applied to most types of biological data: microarray data, protein interaction networks, gene regulatory networks, DNA sequence and protein structure.

This work focuses essentially on microarray data. Microarray technology is a tool to measure, simultaneously and in a single experiment, levels of thousands of gene expression in many varied conditions. This new technology has been very useful for many biological and medical applications. It facilitates data analysis which represents an essential task to extract relevant biological knowledge from large data masses. As stated by Kenyon et al. (2002), this technology has reached a certain maturity. Indeed, many improvements have been made in order to better understand the disease (Berrar et al. 2003), validate the quality of the hybridizing target (Auer et al. 2003), improve their marking (Manduchi et al. 2002) and also to optimize the acquisition and signal processing (Yang et al. 2001).

Given the large quantity of data to analyze, the recourse to the data mining techniques has become essential in order to extract knowledge embedded in these masses of information. Indeed, many of these techniques, such as clustering (Quackenbush 2006), were applied to these data in order to extract relevant knowledge (Wang et al. 2005). This technique allows to group genes with similar behavior under all considered

Communicated by V. Loia.

✉ Ons Maâtouk
ons.maatouk.h@gmail.com

1 LARODEC, Université de Tunis, 92 Boulevard 9 Avril, 1007 Tunis, Tunisia

2 LERIA, Université d'Angers, 2 Bd Lavoisier, 49045 Angers, France

3 LaTICE, Université de Tunis, 92 Boulevard 9 Avril, 1007 Tunis, Tunisia

Published online: 17 July 2018

conditions, which implies that all genes share similar biological functions.

However, during certain cellular processes, gene subsets may have similar behavior only under certain conditions and display independent behaviors for other conditions. According to Yip (2003), the same set of genes may have a similar response to a given environmental stimulus, but each gene may have different functions in other conditions. Similarly, for a set of related conditions such as a set of samples, some genes may have different expression pattern.

Consequently, it is recommended to go beyond traditional clustering prototype and use a more adapted technique, such as the biclustering (Cheng and Church 2000) which is able to group a subset of genes with similar behavior under only a subset of conditions. These resulting submatrices are called biclusters. Unlike clustering which gives a global view of knowledge hidden in the data by creating disjoint clusters, the biclustering gives a local view by creating overlapping biclusters covering generally a portion of the input set. This overlap is planned when every bicluster represents a biological function and the genes belonging to different biclusters can have more than one biological function (Sharan et al. 2006). Indeed, these genes have close biological functions (Quackenbush 2006; Divina and Aguilar-Ruiz 2006; Pontes et al. 2010). In addition, these genes may participate in more than one function (Abdullah and Hussain 2006). Thus, the extraction of these coherent subsets represents a very important task in the analysis of gene expression data.

The biclustering aims to combine simultaneously the rows and the columns of a matrix to obtain consistent, homogeneous and stable biclusters. These biclusters represent a subset of genes with the same behavior under a subset of conditions. Note that each row or each column can participate in one or more biclusters. This data mining technique is broadly used in diverse areas such as pattern discovery (Sang and Sun 2014; Orzechowski et al. 2018), text mining (de Castro et al. 2007; Orzechowski and Boryczko 2016), marketing (Amna and Hermanto 2017; Miao and Zhang 2017), web search (Inbarani and Thangavel 2013; Cachucho et al. 2016) and especially biomedicine (Arikan et al. 2016; Bottarelli et al. 2018), in particular for microarray and gene expression studies (Pontes et al. 2015; Liew 2016; Padilha and Campello 2017; Maind and Raut 2018; Huang et al. 2018).

Formally, the microarray data are represented as a data matrix $M(I, J)$, where I is a set of genes and J a set of conditions, and the cell m_{ij} represents the level of expression of the ith gene relative to the jth condition. A bicluster $B(G, C)$ associated with a data matrix $M(I, J)$ is a submatrix such that $G \subseteq I$ and $C \subseteq J$. The biclustering problem aims to extract biclusters of maximal size that satisfy a coherence constraint. This is expressed as an optimization problem.

The biclustering problem is a highly combinatorial problem with a search space size $O(2^{|I|+|J|})$ (Valente et al. 2013). In addition, in the general case, the biclustering problem is NP-hard (Cheng and Church 2000), which explains why the majority of existing biclustering methods do not guarantee the optimality of their solutions. As mentioned in Valente et al. (2013), biclustering algorithms can be grouped into two classes: systematic algorithms and metaheuristic algorithms.

The biclustering algorithms that adopt a systematic approach include:

- Divide and Conquer Approach: This approach starts with the entire data matrix as the initial bicluster. Then, divided iteratively this bicluster into several biclusters satisfying certain features until the verification of termination criterion such as *two-way splitting* (Hartigan 1975) and *BiMax* (Prelic et al. 2006). This approach is characterized by its swiftness, but it may exclude good biclusters by performing the division before evaluating them.
- Greedy Iterative Search Approach: This approach builds biclusters based on an iterative process which aims to maximize or minimize some functions. This process is based on adding or deleting rows or columns until there is no other possibility of addition or removal. This approach is also characterized by its swiftness. Contrariwise, it can ignore good biclusters. Among the algorithms adopting that approach, there are *OPSM* (Ben-Dor et al. 2002), *ISA* (Ihmels et al. 2004), *BicFinder* (Ayadi et al. 2012a), *BICLIC* (Yun and Yi 2013) and *ICS* (Ahmed et al. 2014).
- Bicluster Enumeration Approach: This approach represents the data matrix as a graph or search tree. Adopting this approach allows the acquisition of the optimal solution. However, it requires a fairly long period of calculations and a large consumption of memory. Among the algorithms using this approach, there are *OP-Cluster* (Liu and Wang 2003), *WLPT* (Trang et al. 2007), *BiBit* (Rodríguez-Baena et al. 2011), *DeBi* (Serin and Vingron 2011), *CE-Tree* (Chen and Chang 2009), *BiMine* (Ayadi et al. 2009), *BiMine+* (Ayadi et al. 2012b) and *SBB* (Hussain and Ramazan 2016).

Biclustering algorithms based on metaheuristic include:

- Neighborhood Search: This approach starts with an initial solution which could be the entire data matrix or only a bicluster. Then each iteration tries to improve the current solution by adding or deleting rows or columns to maximize or minimize certain functions. Contrary to the Greedy Iterative Search Approach, this approach allows the addition of rows or columns that have previously deleted. Among the algorithms using this approach, there are *CC* (Cheng and Church 2000), *Reactive-GRASP* (Dharan and Nair 2009), *CGRASP* Das and Idicula (2010), *BILS* (Ayadi et al. 2010), *PDNS* (Ayadi et al. 2012c) and *LSM* (Maâtouk et al. 2017).

- Evolutionary Algorithm (EA): This approach starts by initializing the population, and each individual represents a possible solution of this population. Thereafter, it assesses each individual based on an evaluation function and selects among them a certain number to produce the new population by applying the crossover and mutation operators, and so on until the verification of stopping criterion. Among the algorithms using this approach, there are *SEBI* (Divina and Aguilar-Ruiz 2006), *SMOB* (Divina and Aguilar-Ruiz 2007; Divina et al. 2012), *CBEB* (Huang et al. 2012), *Evo-Bexpa* (Pontes et al. 2013) and those proposed by Nepomuceno et al. (2015a, b, 2016).
- Hybrid Approaches: This approach combines the evolutionary algorithm and the neighborhood search. Both of these approaches allow the exploration of large search space and provide a compromise between the quality of solution and the execution time. However, it is possible that the research be trapped in a not optimal solution. Among the algorithms using this approach, there are *BiHEA* (Gallo et al. 2009) *SSB* (Nepomuceno et al. 2011) *HMOBI* (Seridi et al. 2011), *PSO-SA-BIC* (Thangavel et al. 2012) and $HMOBI_{ibea}$ (Seridi et al. 2015).

An overview of all these methods shows clearly that the EA approach is widely used to solve the biclustering problem. It can be explained by the fact that EA can explore adequately a large solution space by the use of the population and the appropriate genetic operators. This approach is recognized for its ability to more easily influence the search space to guide search to solutions of a certain type. For example, let us quote the evolutionary algorithm $EBACross$ (Maâtouk et al. 2014). It introduces a crossover method for the biclustering of gene expression data. This method aims to obtain children biclusters with a better quality than their parents. It guarantees the extraction of high-quality biclusters of highly correlated genes which are particularly involved in specific ontology structure. It is therefore easier to integrate particular knowledge by adopting the evolutionary approach and acting on different operators to better manipulate the research space.

The aim of this work is to enhance the performance of the evolutionary algorithm to extract maximal coherent biclusters, by providing different variants of the EA based on different genetic operators. Our choice for the EA approach is also motivated by its conceptual simplicity, its robustness to the dynamic changes and especially for its ability to self-optimization (Fogel 1997). This performance is enhanced by the use of the adequate genetic operators.

In this work, we introduce an evolutionary biclustering algorithm. Like all the evolutionary algorithms, it is essentially based on the genetic operators: selection, crossover and mutation. For each of these operators, two different methods are proposed. The first selection method is based on the parallel approach. It uses four complementary functions: the size function, the mean squared residue function, the average correlation function and the coefficient of variation function, while the second selection method is based on the aggregation of two functions: the size function and the average correlation function. Regarding the crossover method and the mutation method, a random method is used firstly. Then, a new crossover and mutation methods dedicated to the biclustering problem are proposed. Based on the standard deviation function, our proposed crossover method has the purpose to extract child biclusters with a better quality than their parent biclusters, while our proposed mutation method tries to improve the coherence between the genes of the biclusters using the correlation matrix. Indeed, to show the influence of each operator on the performance of the evolutionary biclustering algorithm (EBA), several combinations are made into EBA variants and an extensive experimental study is realized. The experimental study was conducted on real microarray datasets and based on two complementary criteria: statistical criteria and biological criteria. This study assesses the capacity of the different possible variants of EBA to extract maximal coherent biclusters and especially significant biclusters with a biological point of view.

This paper is organized as follows. In Sect. 2, we present our generic evolutionary algorithm for the biclustering problem. Proposed genetic operators and their variants are detailed in Sect. 3. Section 4 is dedicated to an extensive experimental study of the different variants of EBA. Both statistical and biological results are conducted. Conclusions are given in the last section.

2 A generic evolutionary biclustering algorithm (*EBA*)

As stated before, one of the main strengths of EA is its conceptual simplicity. Generally speaking, each evolutionary algorithm adaptation is given by four ingredients:

(1) Evolutionary operators: In this work, we use four operators (selection, crossover, mutation and replacement) to get an efficient compromise between intensification and diversification in the search process.
(2) Encoding method: It could influence the EA behavior. Indeed, this encoding should be related to the problem structure and its basic features.
(3) Fitness functions: They evaluate individuals and measure their "goodness" during the search process to lead to the best individuals.
(4) Archiving technique: It saves the best specimens to enhance future generations.

Algorithm 1: $EBA(Selection, Crossover, Mutation)$

Input: Microarray dataset M, Iteration number $Iter\_Max$
Output: Bicluster set $Rslt$

```
begin
    P ← Initialize Population (M)
    while (Nb_Iter ≠ Iter_Max) do
        IS ← Selection (P)
        Comb ← Crossover (IS)
        Mut ← Mutation (Comb)
        Rslt ← Archiving (Mut)
        P ← Replacement (Mut)
    return (Rslt)
```

To facilitate the study of the different variants of the evolutionary biclustering algorithm, we use the following indexation : *EBA(Selection,Crossover,Mutation)*. The general structure of the proposed evolutionary biclustering algorithm is presented in Algorithm 1. In this algorithm, we do not change the replacement and the archiving strategies and we variate the selection, the crossover and the mutation methods.

The algorithm begins by generating an initial population P. In order to start with reasonable quality biclusters and cover almost the whole matrix, the biclustering algorithm CC, proposed by Cheng and Church (2000), is used for this step. Indeed, this algorithm is recognized for its almost total coverage of genes and conditions and its reasonable results in a quick time.

In the remaining of this section, we present the fitness functions used by the different EBA variants and the solution encoding used to represent the biclusters.

2.1 Fitness functions

The goal of biclustering is to extract maximal biclusters of highly correlated genes. Thus, two properties have to be optimized, the coherence of genes and the size of biclusters. To measure how much an *individual* fits these properties, we consider four fitness functions. The first one is the size, and three of them are complementary measures related to the coherence, as it will be explained below.

2.1.1 Size function

Most of the biclustering algorithms define the size of a bicluster by its number of elements $|G| * |C|$ as in Liu et al. (2009). This function gives more chance to the number of genes to be maximized since the total number of genes is higher than the number of conditions. To be able to choose if we want to give more chance to the number of genes or to the number of conditions to be maximized, we define the size of biclusters by the following function where α is a constant.

$$S(B) = \alpha \frac{|G|}{|I|} + (1-\alpha)\frac{|C|}{|J|} \tag{1}$$

2.1.2 Mean squared residue function

Cheng and Church (2000) proposed *mean squared residue* (MSR) which measures the correlation of a bicluster. A high value of MSR indicates that the bicluster is weakly coherent, while a low value of MSR indicates that it is highly coherent. It is defined as follows:

$$MSR(B) = \frac{\sum_{i=1}^{|G|}\sum_{j=i}^{|C|}(m_{ij} - m_{iC} - m_{Gj} + m_{GC})^2}{|G|\,|C|} \tag{2}$$

where m_{iC} (respectively, m_{Gj}) represents the expression level average of the ith row (respectively, the jth column), m_{GC} corresponds to the expression level average of the bicluster $B(G, C)$, and m_{ij} represents the expression level corresponding to the ith row and the jth column.

A high value of MSR (greater than a certain threshold) indicates that the bicluster is weakly coherent, while a low value of MSR indicates that it is highly coherent. Although the MSR function is good for detecting constant biclusters, it is deficient to detect certain types of biclusters as biclusters with coherent values and those with coherent evolutions (Aguilar-Ruiz 2005; Pontes et al. 2007; Teng and Chan 2008). So to cover this deficiency, we also used the average correlation function.

2.1.3 Average correlation function

Nepomuceno et al. (2010) proposed the average correlation function to evaluate the correlation between genes in each bicluster. They indicate that the proposed function can find biclusters that cannot be found by the algorithms based on MSR. Consequently, algorithms might not find scaling patterns when the variance of gene value is high. The average correlation of the bicluster $B(G, C)$ is defined as follows:

$$\rho(B) = \frac{2}{|G|(|G|-1)}\sum_{i=1}^{|G|}\sum_{j=i+1}^{|G|}\left|\frac{cov(g_i, g_j)}{\sigma_{g_i}\sigma_{g_j}}\right| \tag{3}$$

with:

$$cov(g_i, g_j) = \frac{1}{|C|}\sum_{k=1}^{|C|}(m_{ik} - m_{iC})(m_{jk} - m_{jC}) \tag{4}$$

$$\sigma_{g_i} = \sqrt{\frac{1}{|C|}\sum_{k=1}^{|C|}(m_{ik} - m_{iC})^2} \tag{5}$$

where $cov(g_i, g_j)$ represents the covariance of the rows corresponding to the gene g_i and the gene g_j.
σ_{g_i} (respectively, σ_{g_j}) represents the standard deviations of the rows corresponding to the gene g_i (respectively, the gene g_j). m_{iC} represents the average expression level of the ith row of the bicluster $B(G, C)$. m_{ik} represents the expression level corresponding to the ith row and the kth column.

This measure varies between 0 and 1. If the genes are highly correlated, $\rho(B) = 1$, 0 otherwise. However, the optimization of the average correlation function allows the extraction of all the bicluster kind, even biclusters with constant values on rows and/or columns, whose the MSR function is good for their detection. Therefore, the use of the average correlation function and the MSR function together promotes the extraction of constant biclusters. So, we resort to the coefficient of variation function in order to promote the extraction of biclusters with coherent values and those with coherent evolutions, by diversifying the expression levels in the bicluster and increasing the dispersion around its average.

2.1.4 Coefficient of variation function

Statistically, the coefficient of variation (CV) is used to characterize the variability of the data in a sample by evaluating the percentage of variation relative to its average. The higher the value of the coefficient of variation is, the larger is the dispersion around the average. It allows to compare the variability of several samples that have different average or even which are not expressed in the same units.

By adopting it to the biclustering of microarray data, the coefficient of variation can be considered as a measure to evaluate the variability of genes of a bicluster under all its conditions. This measure is calculated separately for each bicluster and is defined as follows:

$$CV(B) = \frac{\sigma_B}{m_{GC}} \tag{6}$$

where σ_B represents the standard deviation of the bicluster B and m_{GC} corresponds to the average of all the expression levels of the bicluster B.

A bicluster with a high coefficient of variation is a bicluster whose the dispersion of expression levels is high. When a bicluster has a coefficient of variation equal to 0, then it has constant values.

2.2 Bicluster encoding

The majority of existing biclustering algorithms represent the biclusters by a fixed binary string size. This binary string is built by two bit strings, the first one for the genes and the second one for the conditions (Nepomuceno et al. 2009, 2011; Gallo et al. 2009). If the gene or the condition belongs to the bicluster, the string position takes 1, else 0. This representation requires exploration of all genes and conditions of each bicluster. This leads to high consumption of time and memory space. In order to reduce this consumption, we represent the biclusters as a string composed by an ordered gene and condition indices like in de Castro et al. (2007); Seridi et al. (2011).

3 Genetic operators for biclustering

In an evolutionary method, we have to choose appropriate genetic operators that provide a good compromise between intensification and diversification to reach solutions of high quality. Generally speaking, the more random operator is, the more it will lead to diversified region, and the more specific operator to the biclustering problem it is, the more it could *intensify* the quality of the generated bicluster. Hence, for each operator type, we specify a random and a dedicated version to assess later their effectiveness in the search process for good biclusters.

3.1 Selection operators

In order to increase the average quality inside the population, the process of selection removes the low-quality individuals from the population, while the high-quality individuals are reproduced. In this section, we present two different selection methods. The first is a parallel selection ($//S$), while the second is based on an aggregation approach (AgS).

3.1.1 Parallel selection method ($//S$)

This method selects individuals according to the different objectives independently. The idea is to select the best individuals for each fitness function. Thus, several subpopulations are created to combine them and obtain a new population. It was firstly proposed by Schaffer (1985).

For the biclustering problem, we consider the four functions mentioned above: size function, mean squared residue function, average correlation function and coefficient of variation function. This operator is applied to the initial population P. It allows to build the initial individual set IS, which represents the union of the different subpopulations. Each subpopulation comprises the best biclusters of the population P according to either of different fitness functions. To explain this selection method, an example is used.

Example Let us consider a population P composed by six biclusters: B_0, B_1, B_2, B_3, B_4, B_5. The different fitness function values, for each bicluster, are shown in Table 1.

Table 1 Fitness function values of the different biclusters

Biclusters	$S(B)$	$MSR(B)$	$\rho(B)$	$CV(B)$
B_0	0.2	150	0.96	0.62
B_1	0.3	200	1	0.75
B_2	0.23	135	0.83	0.59
B_3	0.1	562	0.79	0.7
B_4	0.25	336	0.52	0.65
B_5	0.29	487	0.69	0.43

First, we define a threshold for each function and create the four subpopulations P_S, P_{MSR}, P_ρ and P_{CV}. Let us consider the size threshold $Th_S = 0.3$, the MSR threshold $Th_{MSR} = 300$, the average correlation threshold $Th_\rho = 0.8$ and the coefficient of variation threshold $Th_{CV} = 0.7$. Let us remember that to have an optimal quality bicluster, we should minimize the MSR value $MSR(B)$ and maximize the size $S(B)$, the average correlation $\rho(B)$ and the coefficient of variation $CV(B)$ of the bicluster B.

P_S comprises the biclusters with a size greater or equal to the size threshold. Only B_1 is found ($P_S = \{B_1\}$). Then, we select the biclusters with a MSR value lower than the MSR threshold. Th_{MSR} contains three biclusters ($P_{MSR} = \{B_0, B_1, B_2\}$). Thereafter, we select the biclusters with an average correlation value higher than the average correlation threshold. The biclusters B_0, B_1 and B_2 have an average correlation value higher than 0.8. So, ($P_\rho = \{B_0, B_1, B_2\}$). We move to the biclusters with a coefficient of variation value higher than the coefficient of variation threshold. Only B_3 is found ($P_{CV} = \{B_3\}$). Thus, the individual set $IS = \{P_S \cup P_{MSR} \cup P_\rho \cup P_{CV}\} = \{B_0, B_1, B_2, B_3\}$.

3.1.2 Aggregation selection method (*AgS*)

The parallel selection is known for its negligence to compromise solutions (Meunier 2002) that incites us to use the aggregation approach (Ishibuchi and Murata 1998). It consists to define a single fitness function F as the weighted sum of the different fitness functions of the original problem. The sum of all weights is equal to 1, and the objectives have to be conflicting (Mitra and Banka 2006) ones to obtain compromising solutions.

For the biclustering problem, since coherence measures (the MSR and the average correlation) are related, we choose two objectives in the aggregation: the size $S(B)$ and the average correlation function $\rho(B)$. Indeed, this choice was motivated by the fact that the optimization of the average correlation function allows the extraction of all the bicluster kind, contrary to the MSR function which is deficient for biclusters with coherent values and those with coherent evolutions (Aguilar-Ruiz 2005; Pontes et al. 2007; Teng and Chan 2008). The fitness function F is defined as follows:

$$F(B) = \beta.S(B) + (1 - \beta).\rho(B) \tag{7}$$

where β is a constant. The higher the F is, the higher is the bicluster quality.

3.2 Crossover operators

The crossover is a basic component in a genetic algorithm. It tries to combine selected individuals to produce one or more offsprings that should inherit parent's features.

Let us consider the two parent biclusters $B_1(G_1, C_1)$ and $B_2(G_2, C_2)$ where $G_1 = \{g_1, g_2, \ldots, g_{n_1}\}$ and $G_2 = \{g'_1, g'_2, \ldots, g'_{n_2}\}$ correspond, respectively, to the gene sets of the two parent biclusters B_1 and B_2.
$C_1 = \{c_1, c_2, \ldots, c_{m_1}\}$ and $C_2 = \{c'_1, c'_2, \ldots, c'_{m_2}\}$ correspond, respectively, to the condition sets of the two parent biclusters B_1 and B_2.

3.2.1 Random order crossover method (*ROX*)

This mechanism is a single-point crossover which combines the biclusters of the IS set in pairs. Each part of bicluster, gene parts and condition parts, is treated separately. Similarly to the combination method used in Seridi et al. (2011), this crossover ensures the order of the genes and the conditions in the new biclusters.

Let us consider the following parent biclusters such as $n_1 \leq n_2$ and $m_1 \leq m_2$:

$$B_1 : g_1\ g_2\ \ldots\ g_i\ \ldots\ g_{n_1}\ //\ c_1\ c_2\ \ldots\ c_j\ \ldots\ c_{m_1}$$
$$B_2 : g'_1\ g'_2\ \ldots\ g'_i\ \ldots\ g'_{n_2}\ //\ c'_1\ c'_2\ \ldots\ c'_j\ \ldots\ c'_{m_2}$$

First, we generate two crossover points g_i and c_j corresponding, respectively, to the gene part and the condition part in B_1 such as $g_1 \leq g_i \leq g_{n_1}$ and $c_1 \leq c_j \leq c_{m_1}$.

Then, we generate g'_i and c'_j the crossover points, respectively, in gene part and condition part in B_2 where $g'_{i-1} \leq g_i \leq g'_i$ and $c'_{j-1} \leq c_j \leq c'_j$.

To build the child biclusters, cut the gene part and the condition part of the first parent bicluster B_1 (respectively, the second parent bicluster B_2), at the crossover points g_i and c_j (respectively, g'_i and c'_j). Afterward, assign the first part $(g_1 \ldots g_i)$ of the first parent bicluster B_1 to the first child bicluster B_{Child_1} and its second part $(g_{i+1} \ldots g_{n_1})$ to the second child bicluster B_{Child_2}. Assign also the first part $(g'_1 \ldots g'_{i-1})$ of the second parent bicluster B_2 to the second child bicluster B_{Child_2} and its second part $(g'_i \ldots g'_{n_2})$ to the first child bicluster B_{Child_1}. The same is applied to the condition part.

$$B_{Child_1} : g_1\ g_2\ \ldots\ g_i\ g'_i\ \ldots\ g'_{n_2}\ //\ c_1\ c_2\ \ldots\ c_j\ c'_j\ \ldots\ c'_{m_2}$$
$$B_{Child_2} : g'_1\ g'_2\ \ldots\ g'_{i-1}\ g_{i+1}\ \ldots\ g_{n_1}\ //\ c'_1\ c'_2\ \ldots\ c'_{j-1}\ c_{j+1}\ \ldots\ c_{m_1}$$

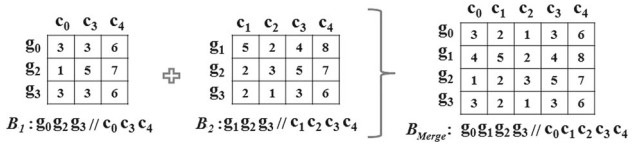


Fig. 1 Creation of the merge bicluster

Example Let us consider the following parent biclusters:

$$B_1 : g_3\ g_4\ g_7\ g_8\ g_{10}\ g_{12}\ g_{13}\ g_{17}\ g_{21}\ //\ c_1\ c_6\ c_9\ c_{11}$$
$$B_2 : g_1\ g_4\ g_5\ g_9\ g_{11}\ g_{15}\ g_{22}\ g_{26}\ g_{27}\ g_{29}\ g_{31}\ //\ c_3\ c_5\ c_7\ c_9\ c_{13}$$

Suppose that $g_i = g_{12}$ and $c_j = c_6$. The gene g_{12}, belonging to the first parent bicluster B_1, can be placed between the genes g_{11} and g_{15}, belonging to the second parent bicluster B_2. The condition c_6, belonging to the first parent bicluster B_1, can be placed between the conditions g_5 and g_7, belonging to the second parent bicluster B_2. Therefore $g'_i = g_{15}$ and $c'_j = c_7$.

Thus, the resulting child biclusters are:

$$B_{Child_1} : g_3\ g_4\ g_7\ g_8\ g_{10}\ g_{12}\ g_{15}\ g_{22}\ g_{26}\ g_{27}\ g_{29}\ g_{31}\ //\ c_1\ c_6\ c_7\ c_9\ c_{13}$$
$$B_{Child_2} : g_1\ g_4\ g_5\ g_9\ g_{11}\ g_{13}\ g_{17}\ g_{21}\ //\ c_3\ c_5\ c_{10}\ c_{11}$$

3.2.2 Biclustering crossover method (*BicX*)

In order to obtain child biclusters with a better quality than their parent biclusters, we propose a crossover method specific for the biclustering algorithms. Unlike the random order crossover, this crossover method considers the two bicluster parts (gene part and condition part) simultaneously. This method is essentially based on four steps.

- Step 1 : Creation of the merge bicluster:

This first step consists to merge the gene sets G_1 and G_2 (respectively, the condition sets C_1 and C_2) of the two parent biclusters B_1 and B_2 into a single set G (respectively, C). This allows to create a new bicluster $B_{Merge}(G, C)$ where $G = G_1 \cup G_2$ correspond to the gene set and $C = C_1 \cup C_2$ correspond to the condition set. An example is presented in Fig. 1.

- Step 2 : Discretization of the merge bicluster:

The second step consists to discretize the merge bicluster B_{Merge}, generated in the previous step. Its goal is to cluster, based on the *standard deviation* function (σ), the conditions with close expression levels, for each gene (row) of the merge bicluster B_{Merge}, independently. The discretization method consists to decompose the merge bicluster B_{Merge} into several vectors. Each vector represents the expression levels of a gene g_i belonging to the gene set G under the condition set C.

This method adopts a heuristic approach. It consists to examine the expression levels ranked in ascending order and assemble their corresponding conditions which the expression levels tend to be very close to their average. It allows to decide each condition belongs to which cluster. For a gene g_i , assigning a condition c_j to a cluster Cl_k depends on the state of this cluster. There are three possible cases:

(1) The current cluster is empty ($Cl_k = \emptyset$): The condition c_j is systematically affected to this cluster Cl_k, by assigning Cl_k as a value to the cell $B_{Discretize_{ij}}$ of the discretized bicluster.
(2) The current cluster contains only one condition ($|Cl_k| = 1$): Compare the standard deviation σ_{g_i} of the gene vector g_i and the standard deviation σ_{Cl_k} of the cluster Cl_k if the condition c_j is added to it.
(3) The current cluster contains several conditions ($|Cl_k| > 1$): Compare the standard deviation σ_{Cl_k} of the cluster Cl_k if the condition c_j is added to it and the standard deviation σ_{Cl_k} of the cluster Cl_k without adding the condition c_j.

The comparisons, carried out in case (2) and case (3), check whether the standard deviation σ_{Cl_k} of the cluster Cl_k improves or deteriorates by adding the condition c_j. If σ_{Cl_k} deteriorates by adding the condition c_j to the cluster Cl_k (Case (3)) or σ_{Cl_k} is lower than σ_{g_i} (Case (2)), the condition c_j is affected to the next cluster Cl_{k+1}, by assigning Cl_{k+1} as a value to $B_{Discretize_{ij}}$. If it is improved, check whether the ascription of the condition c_j to another cluster gives a better standard deviation than that obtained by assigning this condition to the cluster Cl_k.

Here, the importance of the sort of the condition set C according to its expression levels is manifested. It allows to avoid checking with all clusters. Given that the expression levels are ordered, the condition c_j is close either to the previous condition c_{j-1} or to the next condition c_{j+1}. Hence, check only whether the ascription of the condition c_j to the next cluster Cl_{k+1} gives a better standard deviation $\sigma_{Cl_{k+1}}$ than that obtained by its ascription to the cluster Cl_k. Obviously, the condition is affected to the cluster with the lowest standard deviation value, by assigning this cluster as a value to $B_{Discretize_{ij}}$.

Once, this step is repeated for all vectors. The discretized bicluster $B_{Discretize}$ is constructed by assigning to each condition the cluster to which it belongs for each gene. Therefore, this method allows to create a new bicluster $B_{Discretize}$ where the ith row and the jth column correspond, respectively, to the ith gene and the jth condition of the merge bicluster

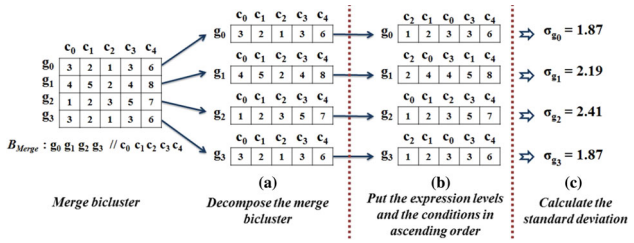


Fig. 2 Decomposition of the merge bicluster B_{Merge} into vectors and calculation of its standard deviations

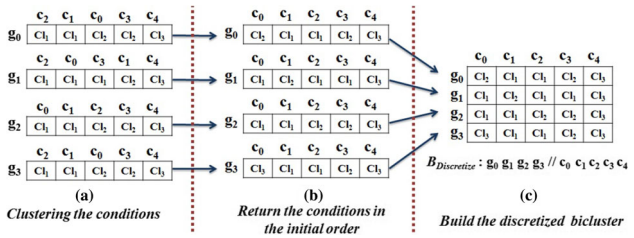


Fig. 3 Clustering of the conditions and construction of the discretized bicluster $B_{Discretize}$

B_{Merge}. The cell $B_{Discretize_{ij}}$ represents the cluster to which the jth condition belongs, for the ith gene.

Let us consider the example presented in Fig. 2, presenting the first part of the discretization method. This method is applied to the merge bicluster B_{Merge} built in step 1 (Fig. 1).

The first part consists to decompose the merge bicluster B_{Merge} into 4 vectors. Each vector represents the expression levels of a gene belonging to the gene set $G = \{g_0, g_1, g_2, g_3\}$ under the condition set $C = \{c_0, c_1, c_2, c_3, c_4\}$ (Fig. 2a).

Then, for the vector corresponding to the gene g_0 (respectively, g_1, g_2 and g_3) calculate its standard deviation σ_{g_0} (respectively, σ_{g_1}, σ_{g_2} and σ_{g_3}) such as in Fig. 2c and sort in ascending order the condition set C according to its expression levels (Fig. 2b).

Afterward, the sorted conditions are clustered based on the standard deviation. Let us take the first vector corresponding to the gene g_0, to explain the clustering method (transition from Fig. 2b to Fig. 3a).

In Fig. 2b, the sorted condition set for the gene g_0 is $\{c_2, c_1, c_0, c_3, c_4\}$, whose the expression levels are, respectively, $\{1, 2, 3, 3, 6\}$. Given that c_2 is the first condition and the current cluster Cl_1 is still empty, this condition is assigned directly to the cluster Cl_1.

Then, the standard deviation of the current cluster σ_{Cl_1} is computed, assuming that the second condition c_1 belongs to it $Cl_1 = \{c_2, c_1\}$. Its standard deviation is equal to $\sigma_{Cl_1} = 0.7$. It is lower than the standard deviation of the vector $\sigma_{g_0} = 1.87$. So check whether the standard deviation of the next cluster, assuming that the condition c_1 belongs to it, $Cl_2 = \{c_1, c_0\}$ is lower than σ_{Cl_1}. The ascription of the condition c_1 to the next cluster Cl_2 gives a standard deviation ($\sigma_{Cl_2} = 0.7$) equal to those of the current cluster σ_{Cl_1}. Therefore, the condition c_1 is assigned to the current cluster $Cl_1 = \{c_2, c_1\}$.

After that, the standard deviation of the current cluster σ_{Cl_1} is computed, assuming that the third condition c_0 belongs to it $Cl_1 = \{c_2, c_1, c_0\}$. Its standard deviation is equal to $\sigma_{Cl_1} = 1$. It is higher than (0.7) the standard deviation of the cluster Cl_1 before assigning the condition c_0 to it. So the condition c_0 is assigned to the next cluster $Cl_2 = \{c_0\}$.

Then, the standard deviation of the cluster σ_{Cl_2} is computed, assuming that the fourth condition c_3 belongs to it $Cl_2 = \{c_0, c_3\}$. Its standard deviation is equal to $\sigma_{Cl_2} = 0$. It is lower than the standard deviation of the vector σ_{g_0}. So check whether the standard deviation of the next cluster, assuming that the condition c_3 belongs to it, $Cl_3 = \{c_3, c_4\}$ is lower than σ_{Cl_2}. The ascription of the condition c_3 to the next cluster Cl_3 gives a standard deviation ($\sigma_{Cl_3} = 2.12$) higher than those of the cluster σ_{Cl_2}. Therefore, the condition c_3 is assigned to the current cluster $Cl_2 = \{c_0, c_3\}$.

After, the standard deviation of the cluster σ_{Cl_2} is computed, assuming that the last condition c_4 belongs to it $Cl_2 = \{c_0, c_3, c_4\}$ and continues in the same way.

Once the conditions of all vectors are clustered (Fig. 3a), return the conditions in their initial order (Fig. 3b). Then, bring the vectors together to build the discretized bicluster $B_{Discretize}$ (Fig. 3c).

- Step 3 :Construction of the variation matrix:

This step consists to build the variation matrix M_{Var} based on the discretized bicluster $B_{Discretize}$. This bicluster presents the condition clusters for each gene. Each cluster gathers the conditions whose the expression levels are close. Since the expression levels were sorted before the clustering in the previous step, the higher the index k of the cluster Cl_k is, the higher are the expression levels of the conditions belonging to this cluster.

For example, in the discretized bicluster $B_{Discretize}$ in Fig. 3c, for the gene g_0, the conditions c_1 and c_2 belong to the cluster Cl_1, while the conditions c_0 and c_3 belong to the cluster Cl_2. So, the expression levels of the gene g_0 under the conditions c_1 and c_2 are lower than its expression levels under the conditions c_0 and c_3. This can be confirmed in Fig. 2. Therefore, to build the variation matrix, the cluster indexes of all condition pairs are compared for each gene. An example is presented in Fig. 4.

The variation matrix M_{Var} shows the expression level variation of the gene set G for each condition pairs of the condition set C. Columns represent the condition pairs $\{(c_1 - c_2), (c_1 - c_3), (c_1 - c_4), ..., (c_{m-1} - c_m)\}$, and rows represent the genes $\{g_1, g_2, ..., g_n\}$ of the discretized bicluster $B_{Discretize}$. Given that the discretized bicluster $B_{Discretize}$ has n rows and m columns, there are $m' = m(m-1)/2$ distinct combinations (condition pairs) between the columns.

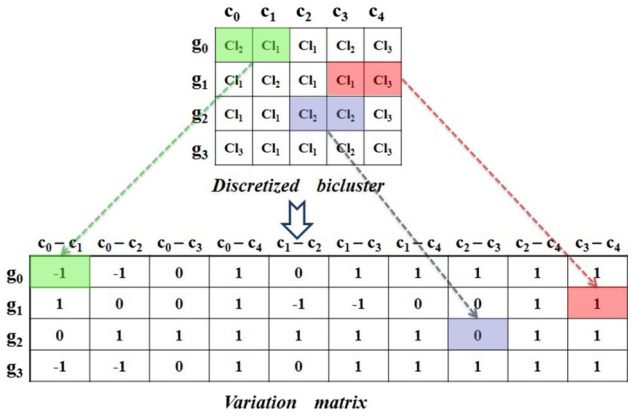


Fig. 4 Construction of the variation matrix

The cell $M_{Var_{ij}}$ represents expression level variation of the ith gene under the jth condition pair $(c_{j_1} - c_{j_2})$. The variation matrix M_{Var} is defined as follows:

$$M_{Var_{ij}} = \begin{cases} -1 : \text{If } B_{Discretize_{ij_1}} > B_{Discretize_{ij_2}} \\ 0 \ : \text{If } B_{Discretize_{ij_1}} = B_{Discretize_{ij_2}} \\ 1 \ : \text{If } B_{Discretize_{ij_1}} < B_{Discretize_{ij_2}} \end{cases} \quad (8)$$

with $i \in [1..n]$, $j \in \left[1..m'\right]$, $j_1 \in [1..m-1]$, $j_2 \in [2..m]$, and $j_2 \geq j_1 + 1$.

- Step 4 :Search of child biclusters:

In order to extract biologically relevant biclusters and knowing that genes presenting similar behavior have close biological functions (Quackenbush 2006; Divina and Aguilar-Ruiz 2006; Pontes et al. 2010), the last step of the $BicX$ crossover consists to extract the child biclusters whose genes follow the same pattern under a condition set. These child biclusters are determined by browsing variation matrix M_{Var} and selecting genes having the same value under the same condition pairs. The pseudocode is presented in Algorithm 2.

Let us consider the example in Fig. 5. First, check whether there are genes in the variation matrix M_{Var} with the same value as the gene g_0 for the condition pair $(c_0 - c_1)$. Only the gene g_3 is found. Now, assign these two genes g_0 and g_3 to the used gene set $G_{Used} = \{\{g_0, g_3\}\}$ and check whether for other condition pairs, the genes g_0 and g_3 have the same value. In this example, the genes g_0 and g_3 have the same value for all condition pairs. So, the first child bicluster contains the genes g_0, g_3 and the conditions c_0, c_1, c_2, c_3 ($B_{Child_1} : g_0\, g_3 \,//\, c_0\, c_1\, c_2\, c_3$).

Then, do the same steps with the following gene g_1. The values are different for all the other genes under the condition pair $(c_0 - c_1)$. So, back to the gene g_0 for the condition pair $(c_0 - c_2)$ and check whether there are any genes with the same value. Only the gene g_3 is found. These two genes already belong to the used gene set $G_{Used} = \{\{g_0, g_3\}\}$. It is the same result obtained with the gene g_0. That will give the same bicluster (B_{Child_1}). Ignore it and go to the next gene g_1 for the condition pair $(c_0 - c_2)$. The values are different for all the other genes under this condition pair.

So, back to the gene g_0 for the condition pair $(c_0 - c_3)$ and check whether there are any genes with the same value. The genes g_1 and g_3 are found. The three genes g_0, g_1 and g_3 do not belong to the used gene set G_{Used}. So, assign it to this set $G_{Used} = \{\{g_0, g_3\}, \{g_0, g_1, g_3\}\}$ and check whether for other condition pairs, the genes g_0, g_1 and g_3 have the same value. These three genes g_0, g_1 and g_3 have the same value for condition pairs $(c_0 - c_3)$, $(c_0 - c_4)$, $(c_2 - c_4)$ and $(c_3 - c_4)$. So, the second child bicluster contains the genes g_0, g_1, g_3 and the conditions c_0, c_2, c_3, c_4 ($B_{Child_2} : g_0\, g_1\, g_3 \,//\, c_0, c_2, c_3, c_4$). Then, go to the next gene and so on, until finding all child biclusters.

Algorithm 2: Search of child biclusters

Input: Variation matrix M_{Var}
Output: Child bicluster $B_{Child}(G_{Child}, C_{Child})$

```
for each jth condition pairs (c_j1, c_j2) with c_j1 ≠ c_j2, c_j1 ∈ C
and c_j2 ∈ C do
    C_Pairs ← {(c_j1, c_j2)}
    for each gene g_i1 ∈ G do
        G_Child ← {g_i1}
        for each gene g_i2 ∈ G\{g_i1} do
            if M_Var_i1j = M_Var_i2j then
                G_Child ← G_Child ∪ {g_i2}
        if G_Child ∉ G_Used then
            G_Used ← G_Used ∪ G_Child
            for each gene g_i ∈ G_Child do
                for each j'th condition pairs (c_j1', c_j2') with
                j' > j, c_j1' ≠ c_j2', c_j1' ∈ C and c_j2' ∈ C do
                    if M_Var_ij = M_Var_ij' then
                        C_Pairs ← C_Pairs ∪ {(c_j1', c_j2')}
            C_Child ← the conditions of C_Pairs
            return (B_Child)
```

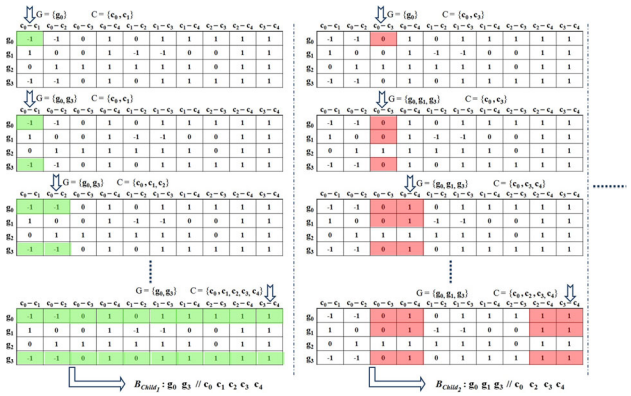


Fig. 5 Search of child biclusters

3.3 Mutation operators

Mutation operator aims to make changes on a current individual. Basically, these changes can be performed randomly or within a specific strategy.

3.3.1 Random mutation method (*RM*)

In order to diversify the genes of the biclusters, a random single-point mutation (RM) is used. This mechanism is applied separately to the gene part and the condition part.

Let us consider $B(G, C)$ a bicluster:

$$B : g_1\ g_2\ \ldots\ g_i\ \ldots\ g_n\ //\ c_1\ c_2\ \ldots\ c_j\ \ldots\ c_m$$

First, we generate two mutation points g_i and c_j corresponding, respectively, to the gene part and the condition part in B such as $g_i \in G$ and $c_j \in C$.

Then, we generate a random gene g'_i and a random condition c'_j to replace, respectively, the chosen gene and condition in B such as $g'_i \notin G$ and $c'_j \notin C$.

Example Let us consider the following bicluster:

$$B : g_3\ g_4\ g_7\ g_8\ g_{10}\ g_{12}\ g_{13}\ g_{17}\ g_{21}\ //\ c_1\ c_6\ c_9\ c_{11}$$

Suppose that g_i=g_8 and c_j=c_6 therefore g'_i=g_{15} and c'_j=c_7. Thus, the resulting child:

$$B_{Child} : g_3\ g_4\ g_7\ g_{10}\ g_{12}\ g_{13}\ g_{15}\ g_{17}\ g_{21}\ //\ c_1\ c_7\ c_9\ c_{11}$$

3.3.2 Biclustering mutation method (*BicM*)

To intensify the quality of the biclusters, a new mutation method dedicated to the biclustering problem is proposed. This method tries to improve the coherence of each gene of these biclusters using the average correlation function. Based on a correlation matrix M_{Corr}, this mutation method computes the number of coherent genes with each gene in each bicluster. If the less coherent gene is coherent with a number of genes lower than the threshold Th_{Corr}, the $BicM$ mutation replaces it with the most coherent gene with those of the bicluster. Otherwise, it adds the most coherent gene to the bicluster.

The cell $M_{Corr_{ij}}$ of this correlation matrix presents the correlation between the gene g_i and the gene g_j. If g_i and g_j represent the same gene, $M_{Corr_{ij}}$ gets -1 Otherwise, calculate the absolute value $\left|\rho_{(g_i,g_j)}\right|$ of the average correlation between the genes g_i and g_j. If it is lower than the correlation threshold Th_{Corr}, $M_{Corr_{ij}}$ gets 0. It gets 1, else.

$$M_{Corr_{ij}} = \begin{cases} -1 : \text{If } i = j. \\ 0 : \text{If } i \neq j \text{ and } \left|\rho_{(g_i,g_j)}\right| < Th_{Corr}. \\ 1 : \text{If } i \neq j \text{ and } \left|\rho_{(g_i,g_j)}\right| \geq Th_{Corr}. \end{cases} \quad (9)$$

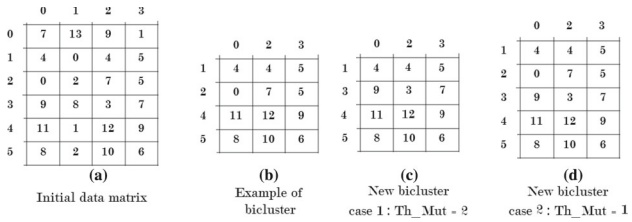


Fig. 6 Biclustering mutation

Table 2 Coherence between each gene pair of the data matrix

Genes	g_0	g_1	g_2	g_3	g_4	g_5
g_0	−1	−0.38	−0.26	−0.04	−0.60	−0.88
g_1	−0.38	−1	0.28	−0.56	0.84	0.29
g_2	−0.26	0.28	−1	−0.92	0.33	0.46
g_3	−0.04	−0.46	−0.92	−1	0.43	−0.57
g_4	−0.60	0.84	0.33	−0.43	−1	0.97
g_5	−0.88	0.29	0.56	−0.57	0.97	−1

Table 3 Correlation matrix

Genes	g_0	g_1	g_2	g_3	g_4	g_5
g_0	−1	0	0	0	1	1
g_1	0	−1	0	1	1	0
g_2	0	0	−1	1	0	1
g_3	0	1	1	−1	1	1
g_4	1	1	0	1	−1	1
g_5	1	0	1	1	1	−1

with $i \in [1..n]$, $j \in [1..n]$.

Example Let us consider the data matrix in Fig. 6a and the bicluster in Fig. 6b.

First, the correlation coefficient between each gene pair under all conditions is computed. The values are presented in Table 2.

Then, according to the Th_{Corr} value, the correlation matrix is defined. Suppose that $Th_{Corr} = 0.40$ which gives Table 3 as a correlation matrix. From this matrix, the coherence between each pair of genes belonging to the bicluster (Fig. 6b) can be checked.

Now, the coherence of the bicluster genes is analyzed based on the correlation matrix (Table 3). The genes g_1 and g_2 are coherent with only one gene of the bicluster (g_1 is coherent with g_4, and g_2 is coherent with g_5), while the genes g_4 and g_5 are coherent with two genes of the bicluster (g_4 is coherent with g_1, g_5, and g_5 is coherent with g_2, g_4).

The two possible cases will be explained. If the Th_{Mut} threshold is equal to 2, there are genes correlated with a number of genes lower than Th_{Mut} as g_1 and g_2 which are

correlated with only one gene. In this case, one of these genes will be replaced. There are four alternatives:

(1) If the gene g_1 is replaced by the gene g_0 :

- The gene g_0 is coherent with two genes g_4 and g_5.
- The gene g_2 is coherent with only one gene g_5.
- The gene g_4 is coherent with two genes g_0 and g_5.
- The gene g_5 is coherent with three genes g_0, g_2 and g_4.

(2) If the gene g_1 is replaced by the gene g_3 :

- The gene g_2 is coherent with only one gene g_5.
- The gene g_3 is coherent with three genes g_2, g_4 and g_5.
- The gene g_4 is coherent with two genes g_3 and g_5.
- The gene g_5 is coherent with three genes g_2, g_3 and g_4.

(3) If the gene g_2 is replaced by the gene g_0 :

- The gene g_0 is coherent with two genes g_4 and g_5.
- The gene g_1 is coherent with only one gene g_4.
- The gene g_4 is coherent with three genes g_0, g_1 and g_5.
- The gene g_5 is coherent with two genes g_0 and g_4.

(4) If the gene g_2 is replaced by the gene g_3 :

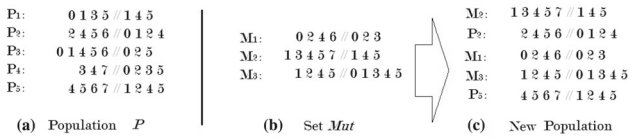
- The gene g_1 is coherent with two genes g_3 and g_4.
- The gene g_3 is coherent with three genes g_1, g_4 and g_5.
- The gene g_4 is coherent with three genes g_1, g_3 and g_5.
- The gene g_5 is coherent with two genes g_3 and g_4.

So, the best possible combination of correlated gene is obtained by replacing the gene g_2 by the gene g_3 (Fig. 6c).

If the Th_{Mut} threshold is equal to 1 and we do not have genes which are correlated with a number of genes lower than Th_{Mut}, the most coherent gene is added to the bicluster. There are two genes not belonging to the group g_0 and g_3. The gene g_0 is coherent with two genes (g_4 and g_5), while the gene g_3 is coherent with four genes (g_1, g_2, g_4 and g_5). So, the gene g_3 will be added to the bicluster (Fig. 6d).

3.4 Archiving method

In order to have a maximum number of high-quality biclusters as final result and avoid losing those biclusters over the different generations, an archiving method is used. This method consists to preserve the best biclusters found in different generations in an external set $Rslt$. These biclusters are selected according to the fitness function $S(B)$, $MSR(B)$, $\rho(B)$ and $CV(B)$ if the parallel selection $//S$ had been used as selection method, otherwise the aggregation function $F(B)$).



(a) Population P (b) Set Mut (c) New Population

Fig. 7 Replacement method

To ensure a greater diversification in the $Rslt$ set and especially to limit the size of this set, we use the Jaccard index. This index measures the overlap between two biclusters, $B_1 = (G_1, C_1)$ and $B_2 = (G_2, C_2)$, in terms of genes and conditions (Madeira et al. 2010). This measure is defined as follows:

$$J(B_1, B_2) = \frac{|B_1 \cap B_2|}{|B_1 \cup B_2|} \tag{10}$$

The biclusters having a Jaccard index greater than 25% are filtered (Madeira et al. 2010), and only the one with a larger size is kept.

3.5 Replacement method

Further to reproduction operators (crossover and mutation), a replacement method is used to determine which individuals will disappear from the population at each generation and which individuals survive to the next generation. The replacement method used removes randomly biclusters from the population and replaces them by those of the Mut set, resulting in the mutation method.

Other strategies are also tested, such as the replacement of the worst biclusters, according to the fitness function, of the population by random biclusters from the set Mut or also by the best biclusters from the set Mut. However, we have not noticed an improvement in the results. So, to save computing time, we opted to keep random replacement method.

Let us consider a population P of 5 biclusters in Fig. 7a and Mut bicluster set including 3 biclusters in Fig. 7b. Three biclusters are chosen randomly from the population P and replaced with three biclusters of the Mut set. Thus, the new population keeps the same size.

4 Experimentation

To facilitate the explanation of the different methods and their influence on the results, we assign symbolic names to the different versions of the **e**volutionary **b**iclustering **a**lgorithm: *EBA(SelectionName,CrossoverName,MutationName)*, by indexing, respectively, the name of the selection method, the

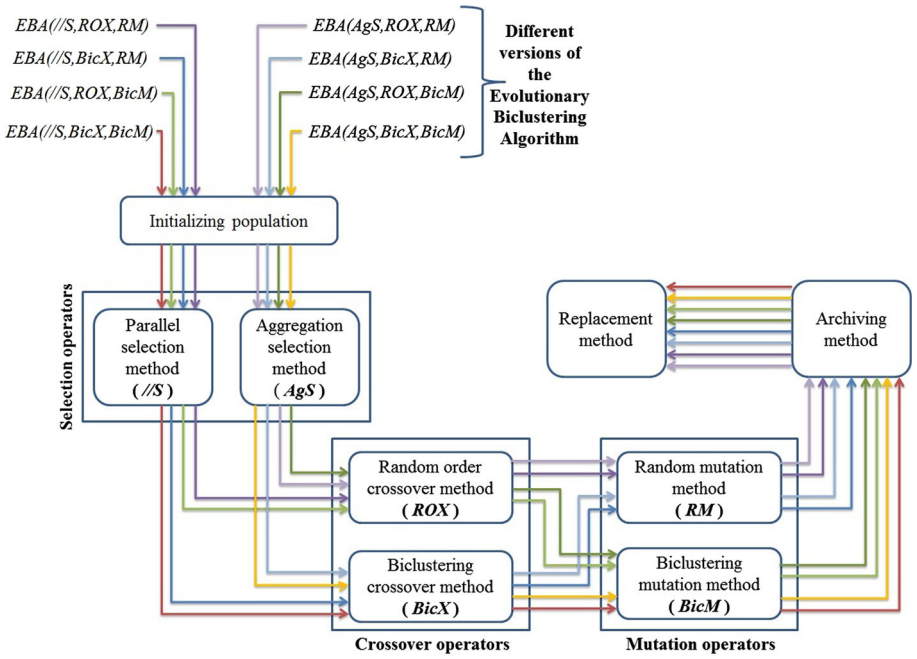


Fig. 8 Different versions of the evolutionary biclustering algorithm (EBA) and their genetic operators

crossover method and the mutation method used. Figure 8 presents the different versions which are :

- *EBA(//S,ROX,RM)*: is published as *EvoBic* in (Ayad et al. 2012d) and based on the parallel selection method $//S$, the random order crossover method ROX and the random mutation method RM.
- *EBA(//S,BicX,RM)*: is published as *EBACross* in (Maâtouk et al. 2014) and based on the parallel selection method $//S$, the crossover method dedicated to the biclustering $BicX$ and the random mutation method RM.
- *EBA(//S,ROX,BicM)*: is based on the parallel selection method $//S$, the random order crossover method ROX and the mutation method dedicated to the biclustering $BicM$.
- *EBA(//S,BicX,BicM)*: is based on the parallel selection method $//S$, the crossover method dedicated to the biclustering $BicX$ and the mutation method dedicated to the biclustering $BicM$.
- *EBA(AgS,ROX,RM)*: is based on the aggregation selection method AgS, the random order crossover method ROX and the random mutation method RM.
- *EBA(AgS,BicX,RM)*: is based on the aggregation selection method AgS, the crossover method dedicated to the biclustering $BicX$ and the random mutation method RM.
- *EBA(AgS,ROX,BicM)*: is based on the aggregation selection method AgS, the random order crossover method ROX and the mutation method dedicated to the biclustering $BicM$.
- *EBA(AgS,BicX,BicM)*: is based on the aggregation selection method AgS, the crossover method dedicated to the biclustering $BicX$ and the mutation method dedicated to the biclustering $BicM$.

These versions allow to understand the usefulness of each operator and their impact on the obtained results.

In order to show the influence of the different methods for each operator on the evolutionary biclustering algorithm performance, an experimental study is realized on their different versions. The results are analyzed and compared with the results of the state-of-the-art biclustering algorithms ISA (Ihmels et al. 2004), $BiMax$ (Prelic et al. 2006), CC (Cheng and Church 2000), $OPSM$ (Ben-Dor et al. 2002), $BiC2PAM$ (Henriques and Madeira 2016), LSM (Maâtouk et al. 2017) and the evolutionary algorithm $HMOBI$ (Seridi et al. 2011), $SEBI$ (Divina and Aguilar-Ruiz 2006), MBA (Ayadi and Hao 2014) and $HMOBI_{ibea}$ (Seridi et al. 2015).

4.1 Datasets

The different versions of our evolutionary biclustering algorithm are tested over two real gene expression datasets:

- *Yeast Cell Cycle*: This dataset contains the expression of 2884 genes under 17 conditions. It has been described by Saeed et al. (1999) and then pretreated by Cheng and Church (2000).
- *Saccharomyces cerevisiae*: This dataset corresponds to a selection of 2993 genes and a collection of 173 different stress conditions. It has been described by Gasch et al. (2000).

4.2 Evaluation criteria

To evaluate the results of the different biclustering algorithms, several criteria are adopted: statistical criteria and biological criteria. These two types of criteria can be considered as being complementary for the validation of each biclustering algorithm.

(1) Statistical criteria:

- Statistical functions: The functions "Size," "Average Correlation" and "Mean Squared Residue" (MSR) are used to measure the quality of the resulting biclusters (Nepomuceno et al. 2010; Seridi et al. 2011).
- Coverage: This criterion is defined as being the total number of cells of the matrix M covered by the resulting biclusters (Cheng and Church 2000; Divina and Aguilar-Ruiz 2007; Liu et al. 2009; Mitra and Banka 2006).

(2) Biological criteria:

- Functional enrichment analysis: Based on the determination of the p value, it measures the quality of the resulting biclusters, by checking whether the bicluster genes have common biological characteristics (Liu and Wang 2007; Nepomuceno et al. 2011; Ayadi and Hao 2014). The biclusters with a p value p lower than 5% are considered as over-represented. This means that the majority of genes in this bicluster have common biological characteristics. The best biclusters have a p value close to 0%.
- Biological signification analysis: It is based on the identification of common functions between genes and also on the identification of the three structures of ontologies describing the genes products in terms of biological process, molecular function and cellular component (Shyama and Mary 2010; Nepomuceno et al. 2011; Balamurugan et al. 2014; Ayadi and Hao 2014).

4.3 Parameter settings

The parameter settings have been fixed after several tests. Every time, we use different parameters to keep those that allow to the better results. The parameters of the fitness function F, used in the aggregation selection method (AgS), are defined by $\beta = \frac{1}{2}$. This gives the same chance to the size function and the average correlation function to be maximized. The size function constant is fixed to $\alpha = \frac{1}{2}$ which gives the same chance to the gene number and the condition number to be maximized. The different versions of the proposed algorithm have been implemented on a PC with an Intel Core i7-3537U processor and 8GB real memory. Run times for some versions can reach one hour.

4.4 Statistical results

This section does not only present the statistical results of the different versions of the EBA but also compare them to several state-of-the-art biclustering algorithms. The statistical quality of the extracted biclusters is defined by the functions: Size, Average correlation, MSR and Coverage.

4.4.1 Size function

This function (Eq. 1) is based on the number of genes and the number of conditions of the extracted biclusters. It can be seen, in Figs. 9, 10 and 11, that the versions using random crossover method (ROX) and random mutation method (RM) allow to extract the largest size biclusters for the *Yeast*

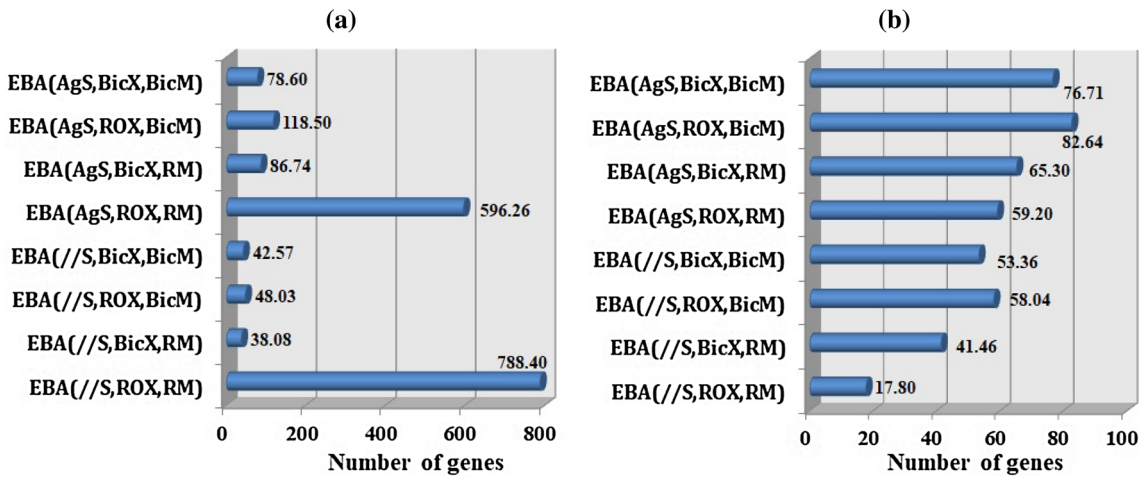


Fig. 9 Average gene numbers of the biclusters extracted by the different versions of the EBA for the datasets: **a** *Yeast Cell Cycle* and **b** *Saccharomyces cerevisiae*

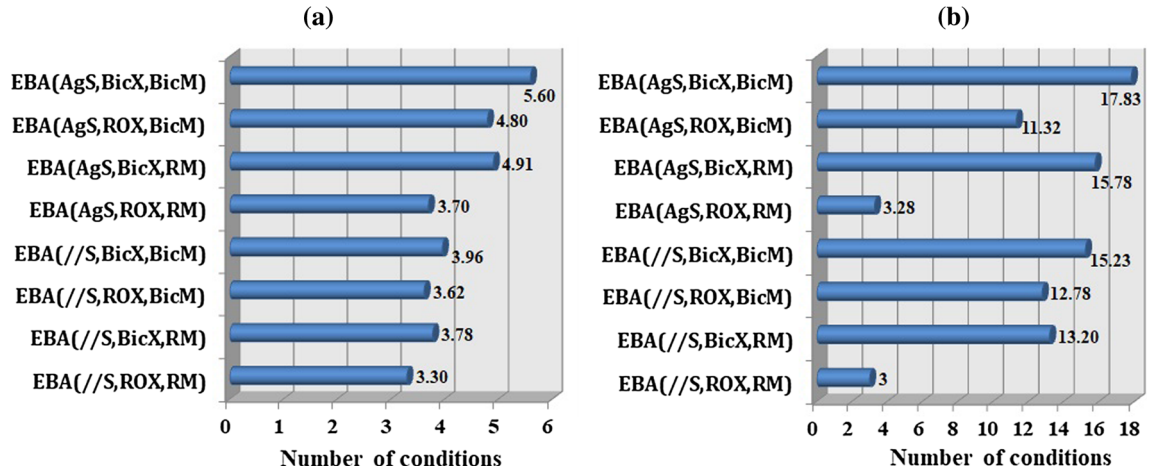


Fig. 10 Average condition numbers of the biclusters extracted by the different versions of the EBA for the datasets: **a** *Yeast Cell Cycle* and **b** *Saccharomyces cerevisiae*

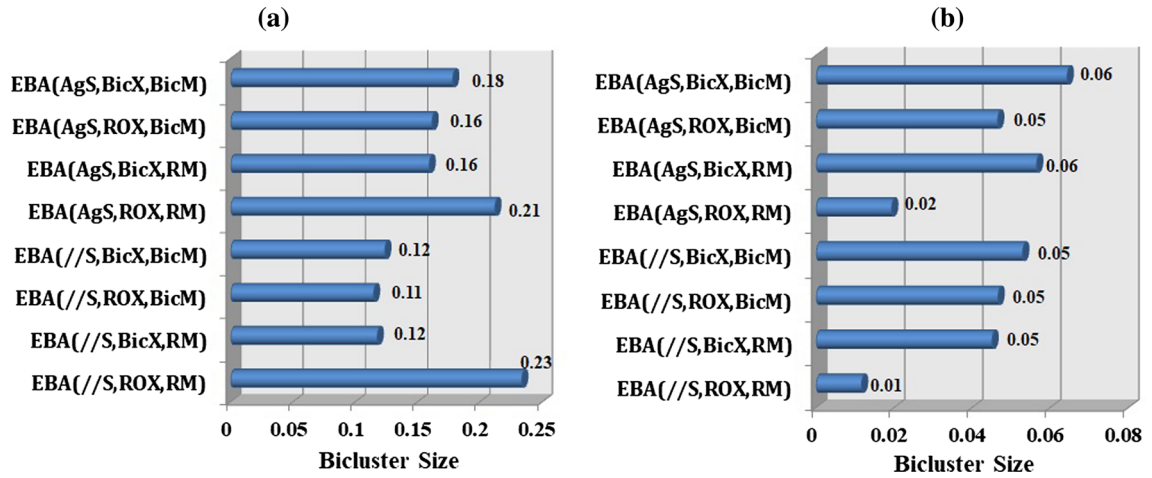


Fig. 11 Average size of the biclusters extracted by the different versions of the *EBA* for the datasets: **a** *Yeast Cell Cycle* and **b** *Saccharomyces cerevisiae*

Cell Cycle dataset, while, for the *Saccharomyces cerevisiae* dataset, the largest ones are extracted by the versions using the crossover ($BicX$) and mutation ($BicM$) methods dedicated to the biclustering.

Moreover, it can be noted that the results of the versions using random crossover (ROX) and mutation (RM) methods decreased considerably for the *Saccharomyces cerevisiae* dataset compared with those obtained for the *Yeast Cell Cycle* dataset.

For the *Saccharomyces cerevisiae* dataset, these two versions recorded very low values, essentially for the number of genes. However, using at least one of the proposed methods of crossover ($BicX$) and mutation ($BicM$), the results are not too different for both datasets.

It can be explained by the fact that the expression levels have strictly positive integer values for the *Yeast Cell Cycle* dataset, while those of the *Saccharomyces cerevisiae* dataset are real numbers including negative values. These results demonstrate the stability of the performance of the versions using the proposed crossover ($BicX$) and/or mutation ($BicM$) method, for different datasets and different types of expression levels.

In addition, the use of the aggregation selection method (AgS) improved the number of genes and conditions, especially when it is used with crossover or mutation method dedicated for biclustering. These results can be predicated since the parallel selection method ($//S$) is based on four functions (including the size function), while the aggregation selection method (AgS) relies only on two functions (size and average correlation). So, it can be said that AgS method gives more importance to bicluster size compared with $//S$ method.

Thus, we can consider that $EBA(//S, ROX, RM)$ (respectively, $EBA(AgS, BicX, BicM)$) has almost the best results among the other versions in terms of bicluster sizes for *Yeast Cell Cycle* (respectively, for the *Saccha-*

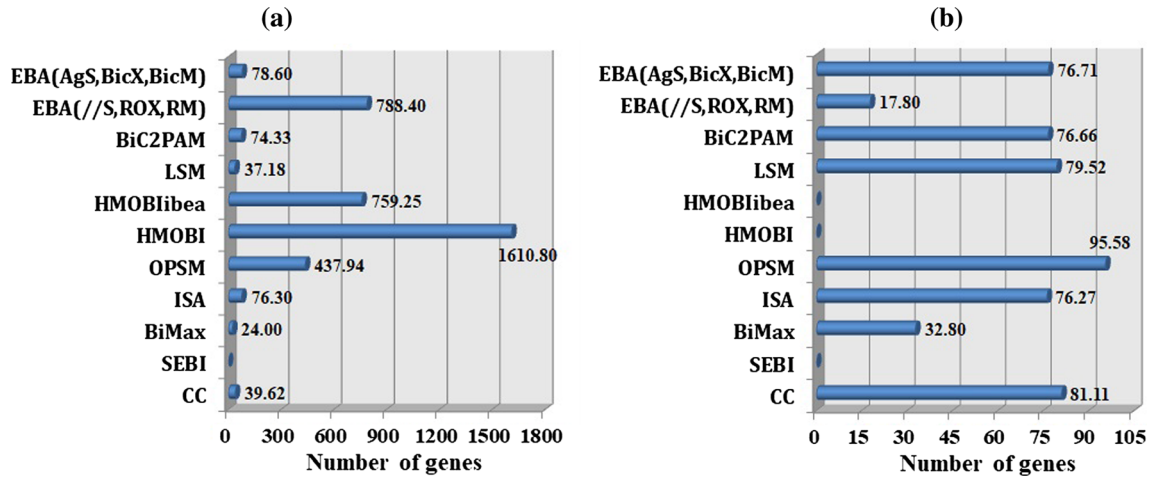


Fig. 12 Average gene numbers of the biclusters extracted by several state-of-the-art biclustering algorithms for the datasets: **a** *Yeast Cell Cycle* and **b** *Saccharomyces cerevisiae*

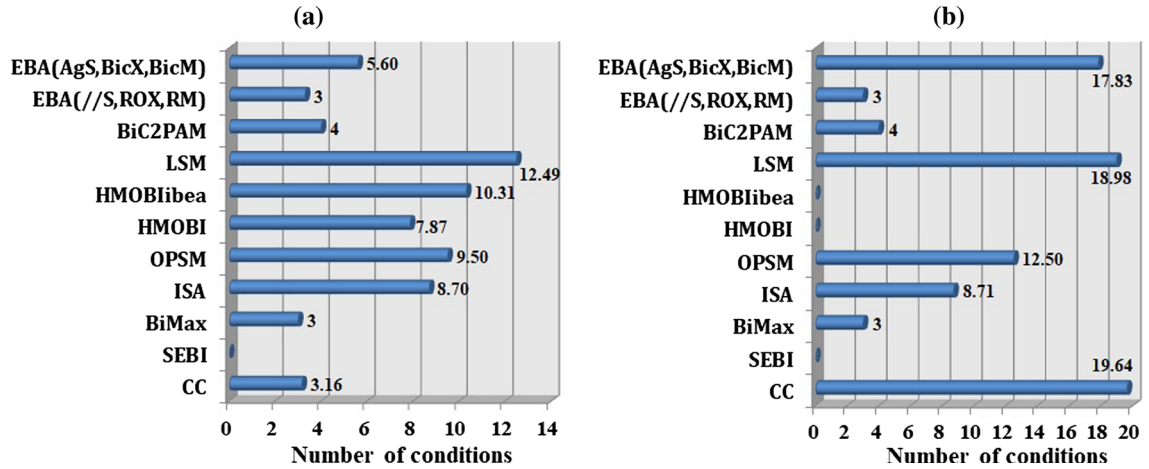


Fig. 13 Average condition numbers of the biclusters extracted by several state-of-the-art biclustering algorithms for the datasets: **a** *Yeast Cell Cycle* and **b** *Saccharomyces cerevisiae*

romyces cerevisiae) dataset. So, the results of these two versions are compared with those of several state-of-the-art biclustering algorithms. Figures 12, 13 and 14 represent, respectively, the average value of the number of genes, the number of conditions and the sizes of the biclusters extracted by some state-of-the-art biclustering algorithms for the *Yeast Cell Cycle* and *Saccharomyces cerevisiae* datasets.

It can be seen that the extracted biclusters are not very large. Indeed, for the *Yeast Cell Cycle* dataset, the largest bicluster is extracted by the $HMOBI_{ibea}$ algorithm. It has a size equal to 0.43 with 759.25 genes and 10.31 conditions. For the *Saccharomyces cerevisiae* dataset, the largest bicluster is extracted by the CC algorithm and has a size equal to 0.07 with 81.11 genes and 19.64 conditions.

Concerning the different versions of our proposed algorithm, $EBA(//S, ROX, RM)$ and $EBA(AgS, BicX, BicM)$ have moderate values for the *Yeast Cell Cycle* dataset. They are not among the algorithms with the largest biclusters but also not the weak ones. For the *Saccharomyces cerevisiae* dataset, the values are rather close and $EBA(AgS, BicX, BicM)$ is among the algorithms with larger biclusters.

4.4.2 Average Correlation function

This function (Eq. 3) evaluates the correlation between the genes in each bicluster. It can be seen in Fig. 15 that the values are rather close. An improvement is noticed for the versions using the crossover method ($BicX$) and the mutation method ($BicM$) dedicated to the biclustering.

It can also be observed that the use of the crossover method dedicated to the biclustering ($BicX$) and the random mutation method (RM) allows to have an average correlation value better than that obtained using the random

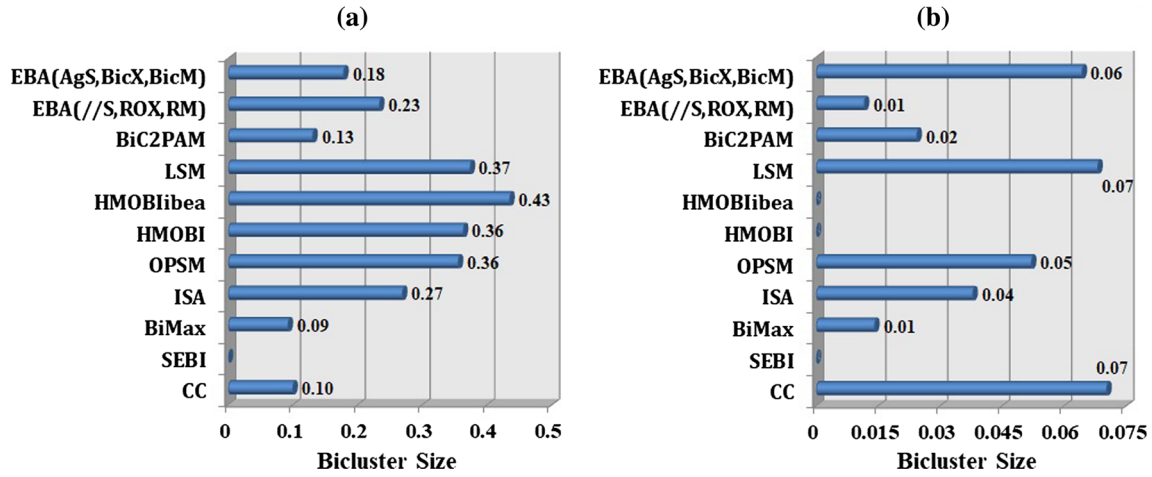


Fig. 14 Average size of the biclusters extracted by several state-of-the-art biclustering algorithms for the datasets: **a** *Yeast Cell Cycle* and **b** *Saccharomyces cerevisiae*

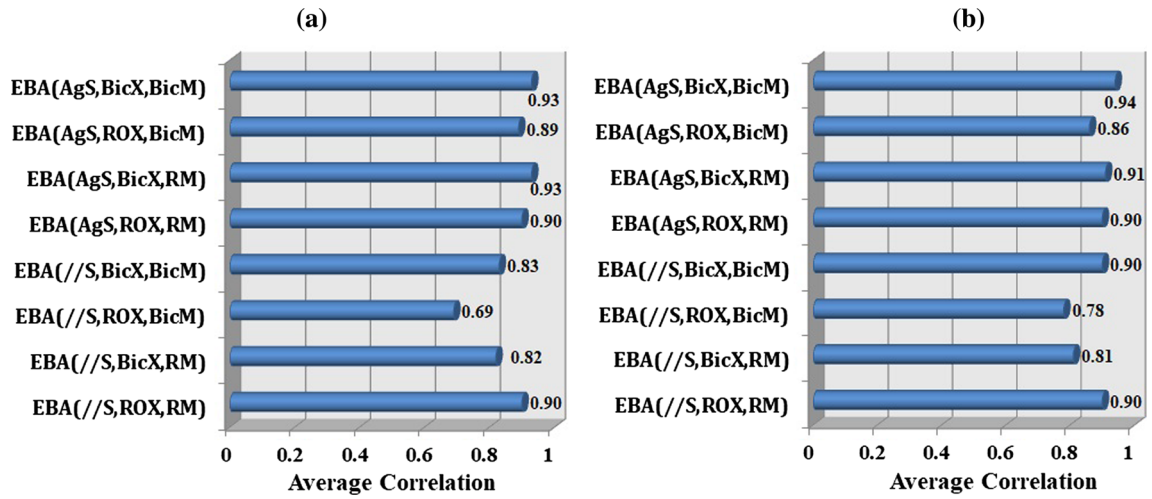


Fig. 15 Average correlation of the biclusters extracted by the different versions of the *EBA* for the datasets: **a** *Yeast Cell Cycle* and **b** *Saccharomyces cerevisiae*

crossover method (ROX) and the mutation method ($BicM$). The values improve slightly using the aggregation selection method (AgS). However, the best average correlation values are recorded using the aggregation selection (AgS), the crossover method ($BicX$) and the mutation method dedicated to the biclustering ($BicM$), together. So, it can be said that these three methods enhance the EBA performance to extract coherent gene biclusters.

These results may be predictable since the AgS selection and the $BicM$ mutation are based on the average correlation function. In addition, the $BicX$ crossover aims to extract biclusters whose genes follow similar patterns under sets of conditions.

The results of this version are compared with those of several state-of-the-art biclustering algorithms and presented in Fig. 16. It can be said that the best values are those of our evolutionary biclustering algorithm (EBA) for the two datasets, precisely the version $EBA(AgS, BicX, BicM)$.

4.4.3 Mean squared residue function

This function (Eq. 2) also measures the bicluster correlation. It can be seen in Fig. 17 that the values are broadly close. It can also be noted that the use of both the crossover method ($BicX$) and the mutation method ($BicM$) dedicated to the biclustering allows the extraction of biclusters with low MSR value.

In addition, the MSR values recorded by the versions which use these two methods together are better than those obtained by the versions using only one of them, essentially when they are used with the aggregation selection method (AgS). For the *Saccharomyces cerevisiae* dataset, the same thing is showed. The version $EBA(AgS, BicX, BicM)$ has

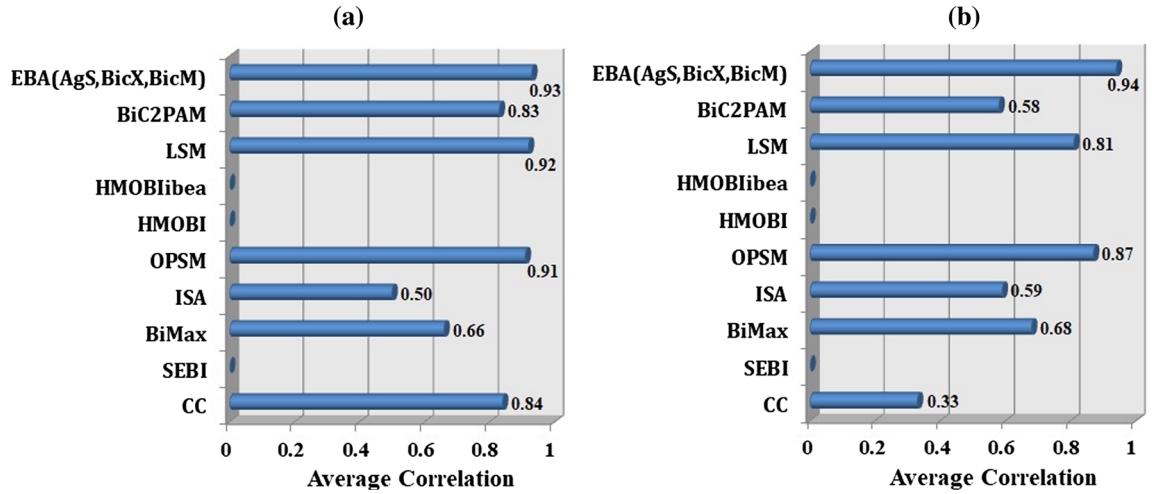


Fig. 16 Average correlation of the biclusters extracted by several state-of-the-art biclustering algorithms for the datasets: **a** *Yeast Cell Cycle* and **b** *Saccharomyces cerevisiae*

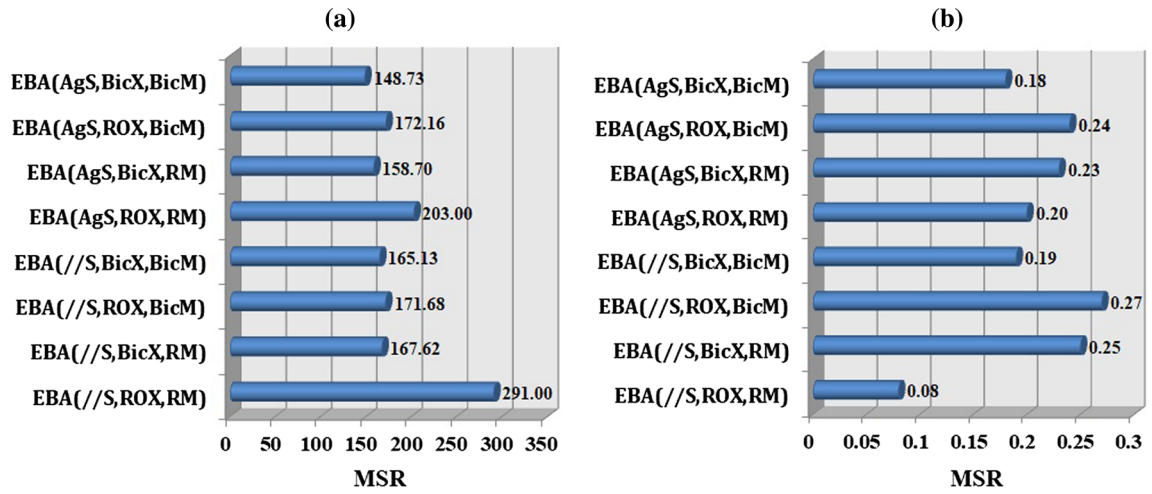


Fig. 17 Mean squared residue of the biclusters extracted by the different versions of the *EBA* for the datasets: **a** *Yeast Cell Cycle* and **b** *Saccharomyces cerevisiae*

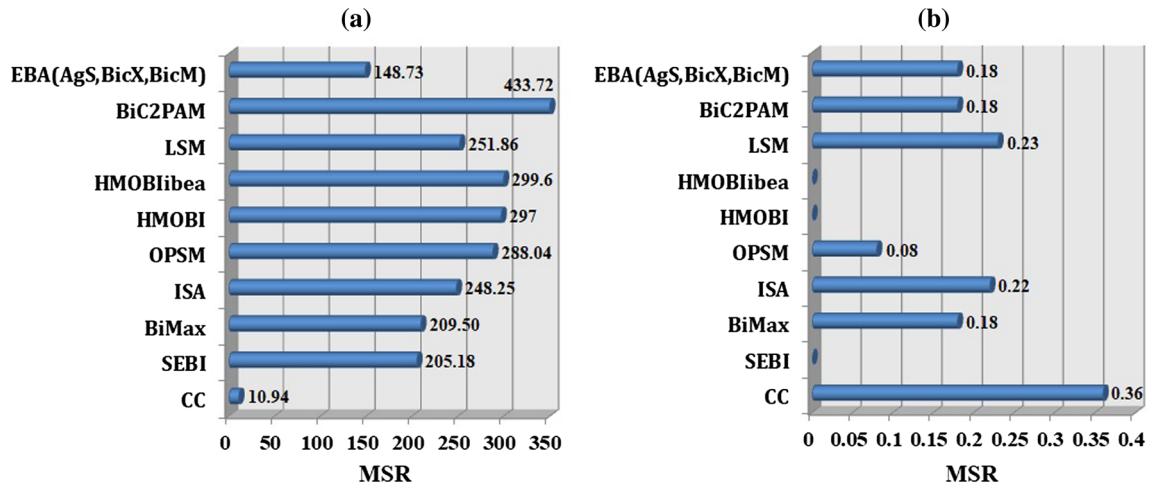


Fig. 18 Mean squared residue of the biclusters extracted by several state-of-the-art biclustering algorithms for the datasets: **a** *Yeast Cell Cycle* and **b** *Saccharomyces cerevisiae*

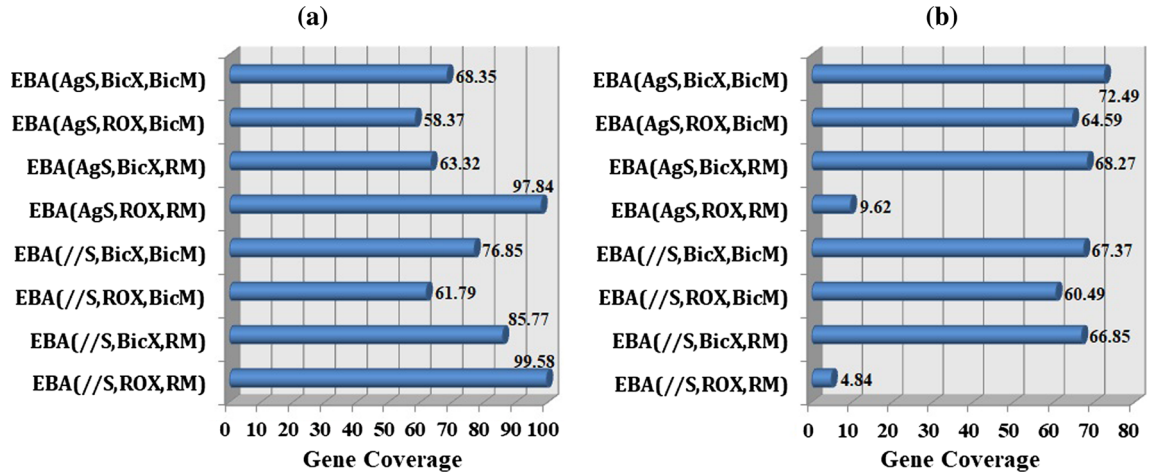


Fig. 19 Gene coverage of the biclusters extracted by different biclustering algorithms for the *Yeast Cell Cycle* and the *Saccharomyces cerevisiae* dataset

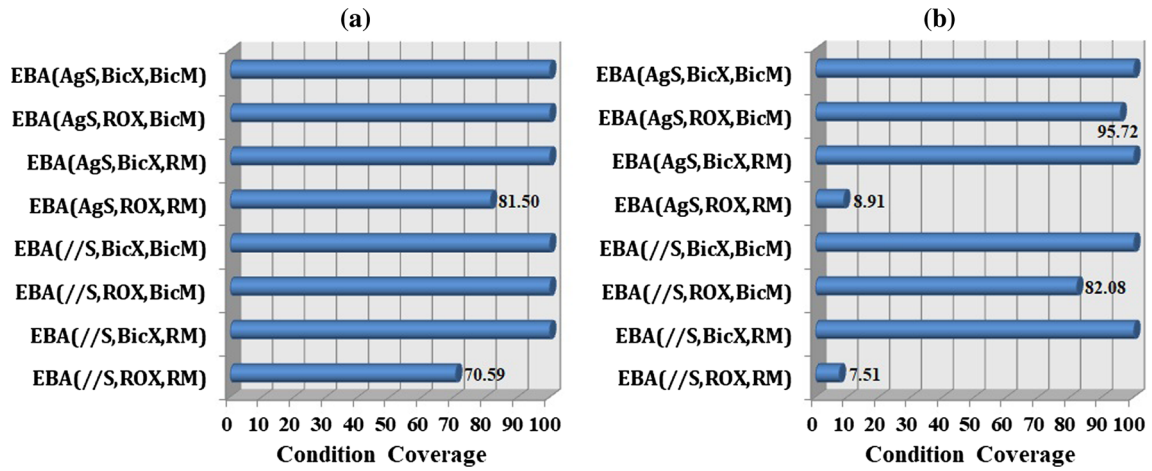


Fig. 20 Condition coverage of the biclusters extracted by different biclustering algorithms for the *Yeast Cell Cycle* and the *Saccharomyces cerevisiae* dataset

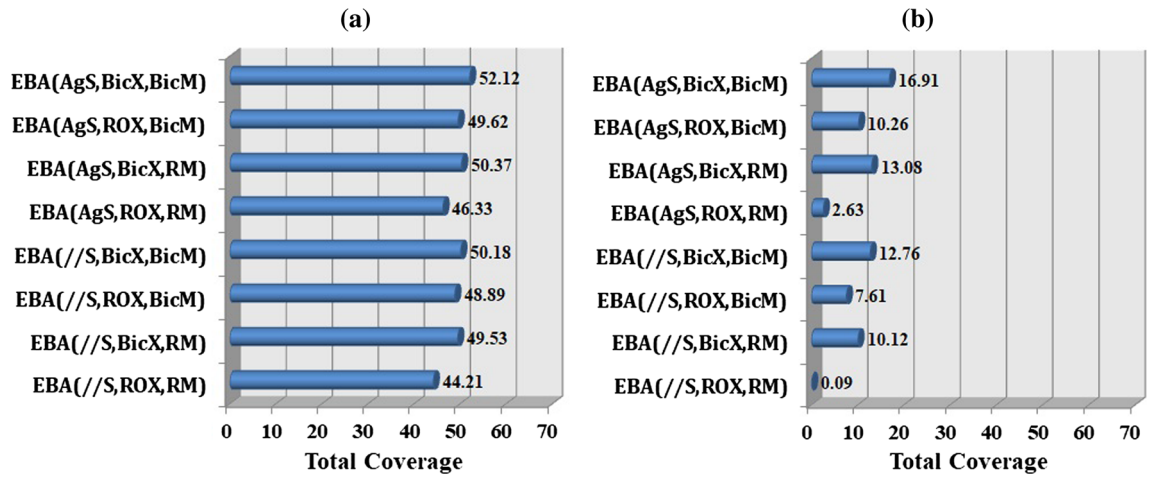


Fig. 21 Total coverage of the biclusters extracted by the different versions of the *EBA* for the datasets: **a** *Yeast Cell Cycle* and **b** *Saccharomyces cerevisiae*

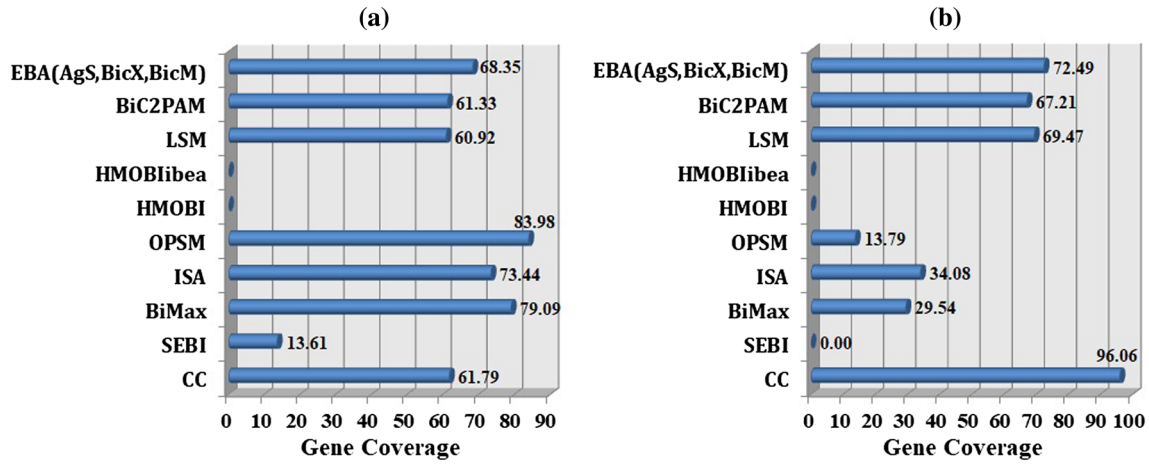


Fig. 22 Gene coverage of the biclusters extracted by different biclustering algorithms for the *Yeast Cell Cycle* and the *Saccharomyces cerevisiae* dataset

the lowest MSR value. So, it can be considered that it has almost the best results among the other versions.

The results of this version are compared with those of several state-of-the-art biclustering algorithms and presented in Fig. 18. $EBA(AgS, BicX, BicM)$ has almost the best MSR values. For the *Yeast Cell Cycle* dataset, only the CC algorithm exceeds it. This low value is due to the fact that the CC algorithm uses the MSR function (allowing to extract constant biclusters) as fitness function. However, for the *Saccharomyces cerevisiae* dataset, the CC algorithm has a high MSR value. This may be explained by the fact that the expression levels of the *Yeast Cell Cycle* have strictly positive integer values, while those of the *Saccharomyces cerevisiae* are real numbers including negative values. In addition, this function is recognized to be good for detecting constant biclusters and deficient to detect other bicluster types (Aguilar-Ruiz 2005; Pontes et al. 2007; Teng and Chan 2008). Thus, it is easier to extract constant biclusters from datasets with integer values. For the *Saccharomyces cerevisiae* dataset, the lowest MSR value is recorded by the $OPSM$ algorithm. However, the EBA result is too close.

4.4.4 Coverage

This criterion is defined as being the total number of cells of the matrix M covered by the resulting biclusters. It can be noticed in Figs. 19, 20 and 21 that most versions have close rates. It can be also shown for both datasets that the versions using the $BicX$ crossover method have better coverage values than the versions which use the random one (ROX). This shows that the proposed crossover method ($BicX$) has a significant influence on increasing the coverage of the obtained biclusters. By trying to determine a maximum of genes presenting the same variation of expression levels under a maximum of conditions, $BicX$ allows to cover a large part of the initial data matrix.

The values improve slightly using the aggregation selection method (AgS). Same using the proposed mutation method ($BicM$), the bicluster coverage increases a little more. So, the best bicluster coverage values are recorded by the version using the aggregation selection method (AgS), the crossover method ($BicX$) and mutation method ($BicM$) dedicated to the biclustering together.

Indeed, $EBA(AgS, BicX, BicM)$ covers 52.12% of the cells of the initial matrix for the *Yeast Cell Cycle* dataset (Fig. 21a) and 16.91% for the *Saccharomyces cerevisiae* dataset (Fig. 21b). It manages to extract biclusters containing 68.35% (Fig. 19a) [respectively, 72.49% (Fig. 19b)] of the genes and all the conditions (Fig. 20a) [respectively, (Fig. 20b)] for the *Yeast Cell Cycle* dataset (respectively, for the *Saccharomyces cerevisiae* dataset).

The results of this version are compared with those of several state-of-the-art biclustering algorithms and presented in Figs. 22, 23 and 24. It can be observed that the results recorded by our evolutionary biclustering algorithm remain the highest ones.

4.5 Biological results

Regarding the biological criteria, they allow to evaluate qualitatively the capacity of an algorithm to extract significant biclusters with a biological point of view, requiring the incorporation of the biological knowledge.

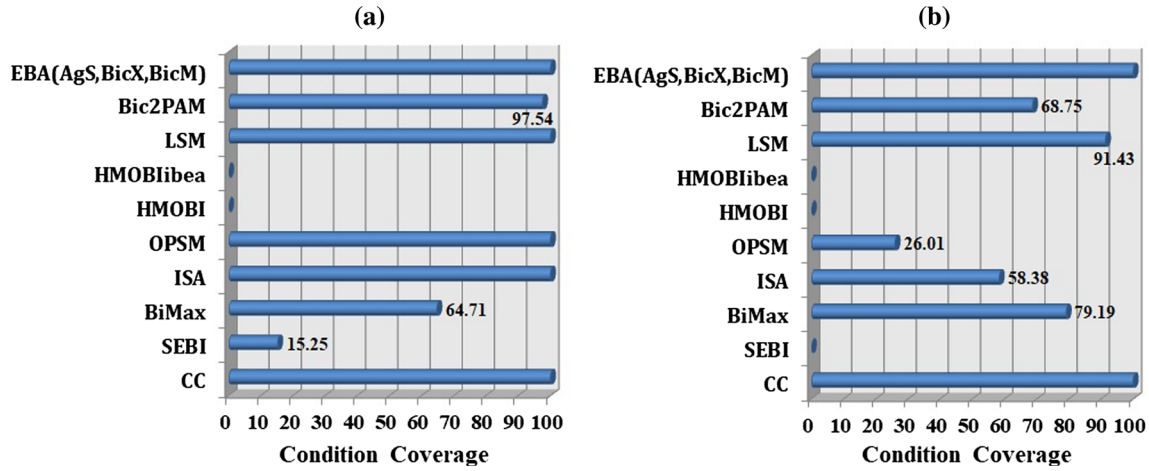


Fig. 23 Condition coverage of the biclusters extracted by different biclustering algorithms for the *Yeast Cell Cycle* and the *Saccharomyces cerevisiae* dataset

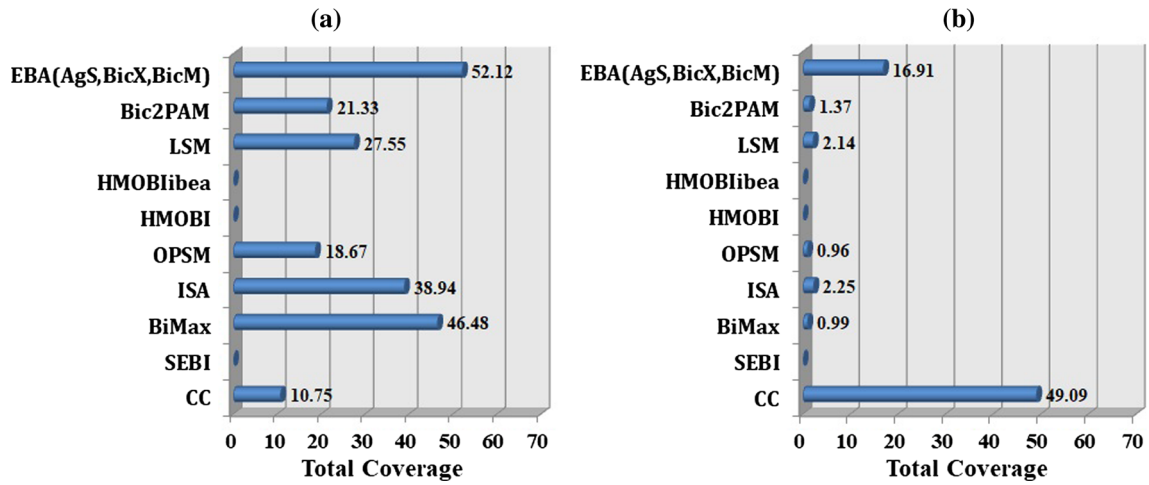


Fig. 24 Total coverage of the biclusters extracted by several state-of-the-art biclustering algorithms for the datasets: **a** *Yeast Cell Cycle* and **b** *Saccharomyces cerevisiae*

4.5.1 Functional enrichment analysis

To assess the statistical significance of the different versions of our evolutionary biclustering algorithm, their results are also compared with those of $Bimax$, CC, ISA, $OPSM$, MBA $BiC2PAM$ and LSM for the *Yeast Cell Cycle* and the *Saccharomyces cerevisiae* datasets. The idea is to compute the adjusted p value of the over-representation of functional classes for a set of genes (Liu and Wang 2007; Nepomuceno et al. 2011; Ayadi and Hao 2014). This p value allows to measure the quality of the extracted bicluster genes. Indeed, the biclusters with a p value p lower than 5% are considered as over-represented. This means that the majority of the genes in this bicluster have common biological characteristics. The best biclusters have a p value close to 0%.

The web tool $FuncAssociate$ (Berriz et al. 2009) is used to evaluate the extracted biclusters. The strategy of (Christinat et al. 2008; Divina and Aguilar-Ruiz 2006; Liu and Wang 2007; Prelic et al. 2006) is followed by realizing the tests only on the best one hundred biclusters. Regarding the other biclustering algorithms, the p values were taken from (Liu and Wang 2007; Nepomuceno et al. 2011).

Figure 25 shows the percentage of enriched biclusters extracted by the different versions of our evolutionary biclustering algorithm (EBA) for several adjusted p value p ($p = 5, 1, 0.5, 0.1, 0.001, 10^{-10}$ and 10^{-50}%) for the *Yeast Cell Cycle* and the *Saccharomyces cerevisiae* datasets.

It can be seen that the EBA versions which use the random crossover method (ROX) and random mutation method (RM) and the CC algorithm have rather low percentage, while the results obtained by the versions which do not use these random methods together are much better. 100% of the biclusters extracted by the versions using at least one of the proposed methods (crossover and/or mutation) are statisti-

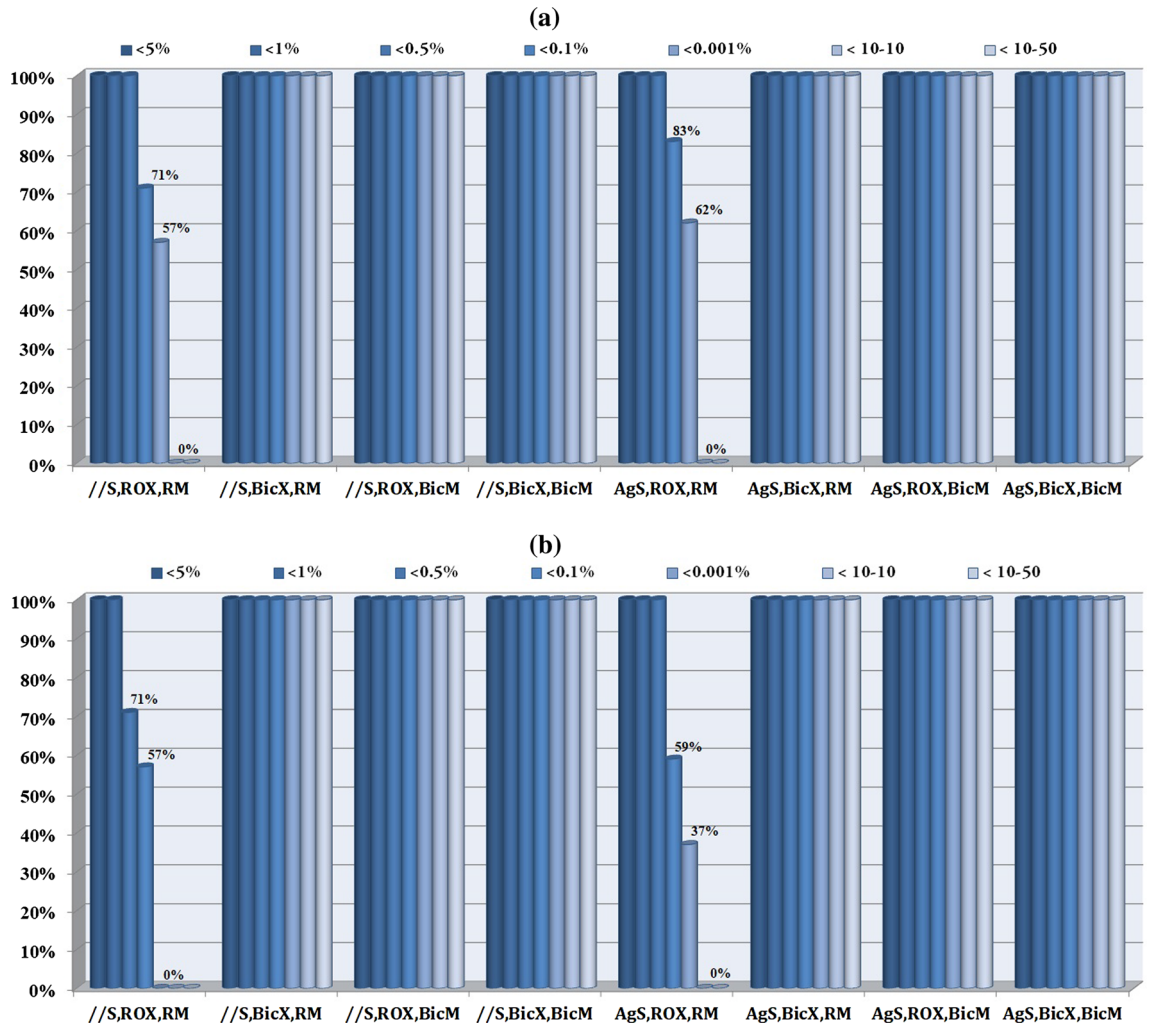


Fig. 25 Percentage of enriched biclusters extracted by the different versions of the *EBA* for different p values for the datasets: **a** *Yeast Cell Cycle* and **b** *Saccharomyces cerevisiae*

cally significant with a p value $p < 10^{-50}\%$ for the two datasets.

Given that $EBA(AgS, BicX, BicM)$ is among the versions having a good percentage of statistically significant biclusters with very low p value and has almost the best values in terms of statistical criteria, the results of this version are compared with those of several state-of-the-art biclustering algorithms. Figure 26 presents the percentage of enriched biclusters extracted by some state-of-the-art biclustering algorithms for different adjusted p value p ($p = 5, 1, 0.5, 0.1, 0.001, 10^{-10}$ and $10^{-50}\%$) for the *Yeast Cell Cycle* and the *Saccharomyces cerevisiae* datasets.

For the *Saccharomyces cerevisiae* (Fig. 26b), 72, 80, 88 and 97% of biclusters, respectively, extracted by $Bimax$, ISA, $OPSM$ and MBA are statistically significant with a p value $p < 0.001\%$. 6 and 15% of biclusters, respectively, extracted by $Bic2PAM$ and LSM are statistically significant with a p value $p < 10^{-50}\%$. Unlike the results found for the *Saccharomyces cerevisiae* dataset, it can be noted that the majority of the results are degraded (Fig. 26a). Only 62, 31, 22 and 92% of the biclusters, respectively, extracted by $Bimax$, ISA, $OPSM$ and MBA are statistically significant with a p value $p < 0.001\%$. 2 and 10% of biclusters, respectively, extracted by $Bic2PAM$ and LSM are statistically significant with a p value $p < 10^{-50}\%$. However, it can be noticed that the EBA outperforms the other biclustering algorithms and preserves its performance for the two datasets. Indeed, 100% of their biclusters are statistically significant with a p value $p < 10^{-50}\%$.

4.5.2 Biological signification analysis

The biological signification of the obtained biclusters can be interpreted based on *gene ontology* (GO) (Ashburner et al. 2000) for the description of the roles of the genes and their products (Robinson et al. 2004). GO provides three ontology

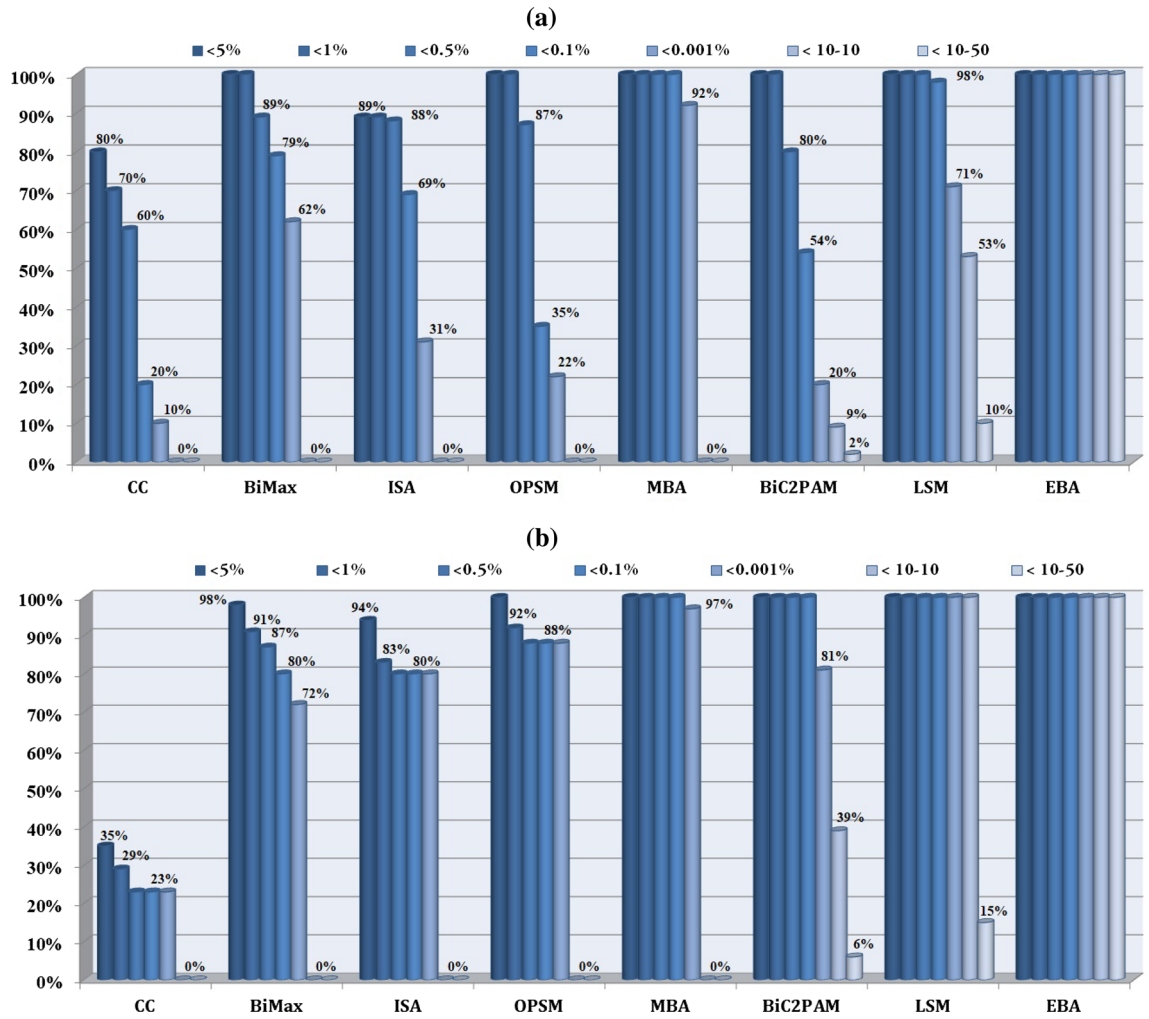


Fig. 26 Percentage of enriched biclusters extracted by several state-of-the-art biclustering algorithms for different p values for the datasets: **a** *Yeast Cell Cycle* and **b** *Saccharomyces cerevisiae*

structures that describe the gene products in terms of biological process, molecular function and cellular component for a given species. For that, the *GOTermFinder* (GOTermFinder 2004; Elizabeth et al. 2004) is used. It is a web tool that seeks significant GO, used to annotate gene products in a given list. In this context, the most significant GO shared of one bicluster extracted by each version of our biclustering algorithm is presented for the *Yeast Cell Cycle* and the *Saccharomyces cerevisiae* datasets.

Tables 4 and 5 show the selected shared GO terms of one bicluster extracted by each version of our evolutionary biclustering algorithm, respectively, for the *Yeast Cell Cycle* and the *Saccharomyces cerevisiae* datasets. The most significant GO terms are those whose the p values are minimal.

Each table includes the gene number and condition number of an extracted bicluster and some of their different shared GO terms for each ontology structures. The values between the parentheses, after each GO term, such as (60%; $7.10\,10^{-4}$) which situated below the cellular component *cytoplasmic translation* of the first bicluster in Table 4, indicate the gene frequency and statistical significance. The gene frequency (60%) shows that 6 genes among 10 genes, of the first bicluster, belong to the *cytoplasmic translation* component, and the statistical significance is provided by the p value p equal to $7.10\,10^{-4}$. Therefore, based on Table 4 and Table 5, it can be deduced that the biclusters extracted by the different versions of our evolutionary biclustering algorithm are biologically relevant for the different gene ontology structures. Indeed, different versions reached to extract biclusters with a very low p value, for the *Yeast Cell Cycle* and the *Saccharomyces cerevisiae* datasets.

4.6 Discussion

This study assesses the capacity of different versions of the evolutionary biclustering algorithm EBA to extract signif-

Table 4 Most significant GO terms of one bicluster extracted by each version of the evolutionary biclustering algorithm EBA for the *Yeast Cell Cycle* dataset

Bicluster size/EBA versions	Cellular component	Molecular function	Biological process
(10×3) / $(//S, ROX, RM)$	*Cytoplasmic translation* (60%; 7.10 10^{-4})	*Nucleobase-containing compound transmembrane transporter activity* (30%; 4, 44.10^{-6})	*Nucleoside transport* (20%; 6, 30.10^{-4})
(13×5) / $(//S, BicX, RM)$	*Nucleolus* (86.7%; 9.4 10^{-32})		
(26×7) / $(//S, ROX, BicM)$	*Large ribosomal subunit* (1.42%; 1.42 10^{-51})		*Protein metabolic process* (58.6%; 1.99 10^{-32})
(105×9) / $(//S, BicX, BicM)$	*Ribosome* (94%; 2.69 10^{-68})	*Structural constituent of ribosome* (60%; 2.58 10^{-66})	*Cytoplasmic translation* (77.8%; 2.79 10^{-82})
(72×11) / (AgS, ROX, RM)		*Structural constituent of ribosome* (63.6%; 8.52 10^{-48})	*Translation* (96.6%; 6.45 10^{-53})
(96×8) / $(AgS, BicX, RM)$	*Intracellular* (21.7%; 6.97 10^{-53})	*Structural molecule activity* (32.77%; 3.37 10^{-81})	*Replication fork* (52%; 4.67 10^{-67})
(342×13) / $(AgS, ROX, BicM)$	*Cytosolic large ribosomal subunit* (65.5%; 1.44 10^{-59})		*Macromolecule biosynthetic process* (98.3%; 6.75 10^{-35})
(58×7) / $(AgS, BicX, BicM)$	*Cytosolic ribosome* (93.1%; 8.68 10^{-85})	*Structural molecule activity* (86.2%; 2.10 10^{-78})	*Cytoplasmic translation* (91.4%; 9.66 10^{-82})

Table 5 Most significant GO terms of one bicluster extracted by each version of the evolutionary biclustering algorithm EBA for the *Saccharomyces cerevisiae* dataset

Bicluster size / EBA versions	Cellular component	Molecular function	Biological process
(6×3) / $(//S, ROX, RM)$	*Golgi vesicle transport* (50%; 8.49 10^{-3})	*Secondary active transmembrane transporter activity* (33.30%; 3.84 10^{-3})	*Trans-Golgi network* (33.30%; 3.95 10^{-3})
(60×7) / $(//S, BicX, RM)$	*Intracellular* (31.67%; 5.2 10^{-6})		
(52×10) / $(//S, ROX, BicM)$	*Chromosome* (63.46%; 5.39 10^{-26})	*Transferase activity* (66.7%; 8.31 10^{-19})	*Negative regulation of biological process* (78.84%; 6.18 10^{-41})
(81×5) $(//S, BicX, BicM)$	*Cytosolic part* (92.59%; 1.45 10^{-54})	*Binding* (48.14%; 7.05 10^{-21})	*Cellular process* (34.58%; 2.83 10^{-37})
(64×13) / (AgS, ROX, RM)	*Intracellular part* (81.25%; 3.70 10^{-30})	*Carbon–sulfur lyase activity* (29.68%; 7.44 10^{-16})	
(79×13) / $(AgS, BicX, RM)$		*Nucleobase-containing* (78.48%; 3.10 10^{-70})	*Ribosome biogenesis* (54.43%; 5.89 10^{-62})
(48×9)/$(AgS, ROX, BicM)$	*Cell* (43.75%; 5.17 10^{-85})	*Nucleoside transmembrane transporter activity* (72.91%; 2.37 10^{-56})	*Regulation of mitotic cell cycle* (35.41%; 9.02 10^{-28})
(36×15) / $(AgS, BicX, BicM)$	*Intracellular organelle* (80.55%; 2.63 10^{-98})	*Nucleobase transmembrane transporter activity* (30.55%; 4.50 10^{-75})	*Regulation of biological process* (69.44%; 8.56 10^{-84})

icant biclusters. It allows to compare its performance by varying the operators (selection, crossing and mutation) in order to retain the best combination.

Overall, it can be noticed that $EBA(AgS, BicX, BicM)$ recorded the best results for almost all criteria. It has the best

Table 6 Comparative table presenting the best biclustering algorithm according to different evaluation criteria for the *Yeast Cell Cycle* and *Saccharomyces cerevisiae* datasets

	$BiMax$	ISA	$OPSM$	$SEBI$	$HMOBI$	$HMOBI_{ibea}$	LSM	MBA	$Bic2PAM$	$EBA\,(AgS, BicX, BicM)$
Size						Best(Y)	Best(S)			
Average correlation										Best(Y,S)
MSR			Best(Y,S)							Best(Y)
Coverage										Best(Y,S)
% enriched biclusters										Best(Y,S)

values in terms of gene correlation (average correlation and MSR), coverage and enriched biclusters for both datasets.

Table 6 summarizes the previously presented results and compares the performance of this obviously best version of the EBA with several state-of-the-art biclustering algorithms for different criteria. It designates the version presenting the best results for each of these criteria. The word "Best" indicates the biclustering algorithm which gives the best values. In parenthesis, the gene expression dataset for which these results are obtained is: "Y" for *Yeast Cell Cycle* and "S" for *Saccharomyces cerevisiae*. It can be clearly seen that the EBA recorded the best results for almost all the assessment criteria.

Thus, it can be concluded that the $EBA(AgS, BicX, BicM)$ version which uses the aggregation selection (AgS), the crossover method ($BicX$) and the mutation method ($BicM$) together, has almost the best known results among the other versions and competes well with the several other biclustering methods.

5 Conclusion

In this paper, several versions of an evolutionary biclustering algorithm (EBA) are proposed for the DNA microarray data. The different versions aim to extract correlated gene biclusters with maximal size and biological relevance. Like all the evolutionary algorithms, the EBA is essentially based on the genetic operators: selection, crossover and mutation. For each operator, two different methods are proposed.

The first selection method ($//S$) is based on the parallel approach. It uses four complementary functions: the size function, the mean squared residue function, the average correlation function and the coefficient of variation function, while the second selection method (AgS) is based on the aggregation of two functions: the size function and the average correlation function. Indeed, each function is used separately; then, the size function and the average correlation function are aggregated to show the influence of the aggregation on the performance of the EBA. In fact, these two functions are conflicting, but their aggregation was done to optimize these conflict objectives simultaneously and to improve the first objective without decreasing the second one.

Regarding the crossover method and the mutation method, a random method is used firstly (ROX and RM). Then, the crossover method ($BicX$) and the mutation method ($BicM$) dedicated to the biclustering algorithm are proposed. Based on the standard deviation function, the crossover method ($BicX$) has the purpose to extract child biclusters with a better quality than their parent biclusters, while the mutation method ($BicM$) tries to improve the coherence between the genes of the biclusters using the correlation matrix.

To assess the performance of the different versions of the evolutionary biclustering algorithm (EBA) in terms of statistical and biological significance and to show the influence of each method on their performance, an experimental study was done on two real microarray datasets. These experimentations show that the different versions allow to extract maximal high-quality biclusters of highly correlated genes and especially significant biclusters with a biological point of view and especially that the $EBA(AgS, BicX, BicM)$ version, which uses the selection method (AgS) based on the aggregation function, the crossover method ($BicX$) and the mutation method ($BicM$) dedicated to the biclustering. This version has almost the best results among the other versions and competes well with the several other biclustering algorithms.

For future works, we could benefit from the help of biologists to integrate biological knowledge in the research process. This will refine the search mechanisms and improve the quality of the extracted biclusters.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Abdullah A, Hussain A (2006) A new biclustering technique based on crossing minimization. Neurocomputing 69(16–18):1882–1896. https://doi.org/10.1016/j.neucom.2006.02.018

Aguilar-Ruiz JS (2005) Shifting and scaling patterns from gene expression data. Bioinformatics 21(20):3840–3845. https://doi.org/10.1093/bioinformatics/bti641

Ahmed HA, Mahanta P, Bhattacharyya DK, Kalita JK (2014) Shifting-and-scaling correlation based biclustering algorithm. IEEE/ACM Trans Comput Biol Bioinform 11(6):1239–1252. https://doi.org/10.1109/TCBB.2014.2323054

Amna AR, Hermanto A (2017) Implementation of BCBimax algorithm to determine customer segmentation based on customer market and behavior. In: Proceedings of the 4th international conference on computer applications and information processing technology (CAIPT'17), pp 1–5. https://doi.org/10.1109/CAIPT.2017.8320694

Arikan SDO, Iyigun C (2016) A supervised biclustering optimization model for feature selection in biomedical dataset classification. In: Proceedings of the data mining and big data, first international conference, (DMBD'16), pp 196–204. https://doi.org/10.1007/978-3-319-40973-3_19

Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, Davis AP, Dolinski K, Dwight SS, Eppig JT, Harris MA, Hill DP, Issel-Tarver L, Kasarskis A, Lewis S, Matese JC, Richardson JE, Ringwald M, Rubin GM, Sherlock G (2000) Gene ontology: tool for the unification of biology. Nat Genet 25:25–29. https://doi.org/10.1038/75556

Auer H, Lyianarachchi S, Newsom D, Klisovic MI, Marcucci G, Kornacker K (2003) Chipping away at the chip bias: RNA degradation in microarray analysis. Nat Genet 35(4):292–293. https://doi.org/10.1038/ng1203-292

Ayadi W, Hao J (2014) A memetic algorithm for discovering negative correlation biclusters of DNA microarray data. Neurocomputing 145:14–22. https://doi.org/10.1016/j.neucom.2014.05.074

Ayadi W, Elloumi M, Hao J (2009) A biclustering algorithm based on a bicluster enumeration tree: application to DNA microarray data. BioData Mining 2:9. https://doi.org/10.1186/1756-0381-2-9

Ayadi W, Elloumi M, Hao J (2010) Iterated local search for biclustering of microarray data. In: Pattern recognition in bioinformatics—5th IAPR international conference (PRIB), pp 219–229. https://doi.org/10.1007/978-3-642-16001-1_19

Ayadi W, Elloumi M, Hao J (2012a) BicFinder: a biclustering algorithm for microarray data analysis. Knowl Inf Syst 30(2):341–358. https://doi.org/10.1007/s10115-011-0383-7

Ayadi W, Elloumi M, Hao J (2012b) BiMine+: an efficient algorithm for discovering relevant biclusters of DNA microarray data. Knowl Based Syst 35:224–234. https://doi.org/10.1016/j.knosys.2012.04.017

Ayadi W, Elloumi M, Hao J (2012c) Pattern-driven neighborhood search for biclustering of microarray data. BMC Bioinform 13(S–7):S11. https://doi.org/10.1186/1471-2105-13-S7-S11

Ayadi W, Maâtouk O, Bouziri H (2012d) Evolutionary biclustering algorithm of gene expression data. In: 23rd international workshop on database and expert systems applications (DEXA), pp 206–210. https://doi.org/10.1109/DEXA.2012.46

Balamurugan R, Natarajan AM, Premalatha K (2014) Comparative study on swarm intelligence techniques for biclustering of microarray gene expression data. Int J Comput Electr Autom Control Inf Eng 8(2):333–339

Ben-Dor A, Chor B, Karp RM, Yakhini Z (2002) Discovering local structure in gene expression data: the order-preserving submatrix problem. In: Proceedings of the sixth annual international conference on computational biology, RECOMB 2002, Washington, DC, USA, April 18–21, 2002, pp 49–57. https://doi.org/10.1145/565196.565203

Berrar DP, Dubitzky W, Granzow M (2003) A practical approach to microarray data analysis. Kluwer Academic Publishers, Dordrecht. https://doi.org/10.1007/b101875

Berriz GF, Beaver JE, Cenik C, Tasan M, Roth FP (2009) Next generation software for functional trend analysis. Bioinformatics 25(22):3043–3044. https://doi.org/10.1093/bioinformatics/btp498

Bottarelli L, Bicego M, Denitto M, Di Pierro A, Farinelli A, Mengoni R (2018) Biclustering with a quantum annealer. Soft Comput. https://doi.org/10.1007/s00500-018-3034-z

Cachucho R, Liu K, Nijssen S, Knobbe AJ (2016) Bipeline: a web-based visualization tool for biclustering of multivariate time series. In: Proceedings of the machine learning and knowledge discovery in databases—European conference, ECMLPKDD'16 , Part III, pp 12–16. https://doi.org/10.1007/978-3-319-46131-1_3

Chen J, Chang Y (2009) A condition-enumeration tree method for mining biclusters from DNA microarray data sets. Biosystems 97(1):44–59. https://doi.org/10.1016/j.biosystems.2009.04.003

Cheng Y, Church GM (2000) Biclustering of expression data. In: Proceedings of the eighth international conference on intelligent systems for molecular biology, August 19–23, 2000, La Jolla/San Diego, CA, USA, pp 93–103

Christinat Y, Wachmann B, Zhang L (2008) Gene expression data analysis using a novel approach to biclustering combining discrete and continuous data. IEEE/ACM Trans Comput Biol Bioinform 5(4):583–593. https://doi.org/10.1145/1486911.1486917

Das S, Idicula SM (2010) Application of cardinality based GRASP to the biclustering of gene expression data. Int J Comput Appl 1(18):44–51. https://doi.org/10.5120/384-575

de Castro PAD, de França FO, Ferreira HM, Zuben FJV (2007) Applying biclustering to text mining: an immune-inspired approach. In: Artificial immune systems, 6th international conference (ICARIS), pp 83–94. https://doi.org/10.1007/978-3-540-73922-7_8

Dharan S, Nair AS (2009) Biclustering of gene expression data using reactive greedy randomized adaptive search procedure. BMC Bioinform. https://doi.org/10.1186/1471-2105-10-S1-S27

Divina F, Aguilar-Ruiz JS (2006) Biclustering of expression data with evolutionary computation. IEEE Trans Knowl Data Eng 18(5):590–602. https://doi.org/10.1109/TKDE.2006.74

Divina F, Aguilar-Ruiz JS (2007) A multi-objective approach to discover biclusters in microarray data. In: Genetic and evolutionary computation conference, GECCO 2007, proceedings, London, England, UK, July 7–11, 2007, pp 385–392. https://doi.org/10.1145/1276958.1277038

Divina F, Pontes B, Giráldez R, Aguilar-Ruiz JS (2012) An effective measure for assessing the quality of biclusters. Comput Biol Med 42(2):245–256. https://doi.org/10.1016/j.compbiomed.2011.11.015

Elizabeth BI, Shuai W, Jeremy G, Heng J, David B, Michael CJ, Gavin S (2004) GO: termfinder-open source software for accessing gene ontology information and finding significantly enriched gene ontology terms associated with a list of genes. Bioinformatics 20(18):3710–3715. https://doi.org/10.1093/bioinformatics/bth456

Fogel DB (1997) The advantages of evolutionary computation. In: Biocomputing and emergent computation: proceedings of BCEC97, pp 1–11

Gallo CA, Carballido JA, Ponzoni I (2009) BiHEA: a hybrid evolutionary approach for microarray biclustering. In: Advances in bioinformatics and computational biology, 4th Brazilian symposium on bioinformatics (BSB), pp 36–47. https://doi.org/10.1007/978-3-642-03223-3_4

Gasch AP, Spellman PT, Kao CM, Carmel-Harel O, Eisen MB, Storz G, Botstein D, Brown PO (2000) Genomic expression programs

in the response of yeast cells to environmental changes. Mol Biol Cell 11(12):4241–4257
GOTermFinder (2004) http://db.yeastgenome.org/cgi-bin/go/gotermfinde
Hartigan JA (1975) Clustering algorithms. Wiley, Hoboken
Henriques R, Madeira SC (2016) BiC2PAM: constraint-guided biclustering for biological data analysis with domain knowledge. Algorithms Mol Biol 11(1):2–23. https://doi.org/10.1186/s13015-016-0085-5
Huang Q, Tao D, Li X, Liew AW (2012) Parallelized evolutionary learning for detection of biclusters in gene expression data. IEEE/ACM Trans Comput Biol Bioinf 9(2):560–570. https://doi.org/10.1109/TCBB.2011.53
Huang X, Zhang L, Wang B, Li F, Zhang Z (2018) Feature clustering based support vector machine recursive feature elimination for gene selection. Appl Intell 48(3):594–607. https://doi.org/10.1007/s10489-017-0992-2
Hussain SF, Ramazan M (2016) Biclustering of human cancer microarray data using co-similarity based co-clustering. Expert Syst Appl 55:520–531. https://doi.org/10.1016/j.eswa.2016.02.029
Ihmels J, Bergmann S, Barkai N (2004) Defining transcription modules using large-scale gene expression data. Bioinformatics 20(13):1993–2003. https://doi.org/10.1093/bioinformatics/bth166
Inbarani HH, Thangavel K (2013) Effective web personalisation based on rough biclustering. Int J Granul Comput Rough Sets Intell Syst (IJGCRSIS'13) 3(1):59–84, https://doi.org/10.1504/IJGCRSIS.2013.054127
Ishibuchi H, Murata T (1998) A multi-objective genetic local search algorithm and its application to flowshop scheduling. IEEE Trans Syst Mand Cybern C 28(3):392–403. https://doi.org/10.1109/5326.704576
Kenyon GL, DeMarini DM, Fuchs E, Galas DJ, Kirsch JF, Leyh TS, Moos WH, Petsko GA, Ringe D, Rubin GM, Sheahan LC, National Research Council Steering Committee (US) (2002) Defining the mandate of proteomics in the post-genomics era: Workshop report. http://europepmc.org/books/NBK95348
Liew AWC (2016) Biclustering Analysis of Gene Expression Data Using Evolutionary Algorithms. John Wiley & Sons, Inc., chap 4, 67–95. https://doi.org/10.1002/9781119079453.ch4
Liu J, Wang W (2003) Op-cluster: clustering by tendency in high dimensional space. In: Proceedings of the 3rd IEEE international conference on data mining ((ICDM), pp 187–194. https://doi.org/10.1109/ICDM.2003.1250919
Liu J, Li Z, Hu X, Chen Y (2009) Biclustering of microarray data with MOSPO based on crowding distance. BMC Bioinform. https://doi.org/10.1186/1471-2105-10-S4-S9
Liu X, Wang L (2007) Computing the maximum similarity bi-clusters of gene expression data. Bioinformatics 23(1):50–56. https://doi.org/10.1093/bioinformatics/btl560
Maâtouk O, Ayadi W, Bouziri H, Duval B (2014) Evolutionary algorithm based on new crossover for the biclustering of gene expression data. In: Proceedings of the 9th international conference pattern recognition in bioinformatics (PRIB'14), pp 48–59. https://doi.org/10.1007/978-3-319-09192-1_5
Maâtouk O, Ayadi W, Bouziri H, Duval B (2017) Local search method based on biological knowledge for the biclustering of gene expression data. 7th International workshop on combinations of intelligent methods and applications (CIMA 17) as part of 21st international conference on knowledge-based and intelligent information & engineering systems (KES 17), Marseille, France vol 6(2), pp 65–74
Madeira SC, Teixeira MC, Sá-Correia I, Oliveira AL (2010) Identification of regulatory modules in time series gene expression data using a linear time biclustering algorithm. IEEE/ACM Trans Comput Biol Bioinform 7(1):153–165. https://doi.org/10.1145/1719272.1719289
Maind A, Raut S (2018) Comparative analysis and evaluation of biclustering algorithms for microarray data. In: Perez GM, Mishra KK, Tiwari S, Trivedi MC (eds) Networking communication and data knowledge engineering, vol 4. Springer, Singapore, pp 159–171. https://doi.org/10.1007/978-981-10-4600-1_15
Manduchi E, Scearce LM, Brestelli JE, Grant GR, Kaestner KH, Stoeckert CJJ (2002) Comparison of different labeling methods for 2-channel high-density microarray experiments. Physiol Genomics 10(3):169–179. https://doi.org/10.1152/physiolgenomics.00120.2001
Meunier H (2002) Algorithmes évolutionnaires parallèles pour l'optimisation multi-objectif de réseaux de télécommunications mobiles. PhD thesis, Université des Sciences et Technologies de Lille, France
Miao Y, Zhang H (2017) A biclustering-based lead user identification methodology applied to xiaomi. In: Li X, Xu X (eds) Proceedings of the fourth international forum on decision sciences. Springer, Singapore, pp 865–871. https://doi.org/10.1007/978-981-10-2920-2_80
Mitra S, Banka H (2006) Multi-objective evolutionary biclustering of gene expression data. Pattern Recognit 39(12):2464–2477. https://doi.org/10.1016/j.patcog.2006.03.003
Nepomuceno JA, Lora AT, Aguilar-Ruiz JS (2009) A hybrid metaheuristic for biclustering based on scatter search and genetic algorithms. In: Pattern recognition in bioinformatics, 4th IAPR international conference, PRIB 2009, Sheffield, UK, September 7–9, 2009. Proceedings, pp 199–210. https://doi.org/10.1007/978-3-642-04031-3_18
Nepomuceno JA, Lora AT, Aguilar-Ruiz JS (2010) Evolutionary metaheuristic for biclustering based on linear correlations among genes. In: Proceedings of the 2010 ACM symposium on applied computing (SAC), Sierre, Switzerland, March 22–26, 2010, pp 1143–1147. https://doi.org/10.1145/1774088.1774329
Nepomuceno JA, Lora AT, Aguilar-Ruiz JS (2011) Biclustering of gene expression data by correlation-based scatter search. BioData Min 4(1):3. https://doi.org/10.1186/1756-0381-4-3
Nepomuceno JA, Troncoso A, Nepomuceno-Chamorro IA, Aguilar-Ruiz JS (2015a) Integrating biological knowledge based on functional annotations for biclustering of gene expression data. Comput Methods Program Biomed 119(3):163–180. https://doi.org/10.1016/j.cmpb.2015.02.010
Nepomuceno JA, Troncoso A, Nepomuceno-Chamorro IA, Aguilar-Ruiz JS (2015b) Scatter search-based identification of local patterns with positive and negative correlations in gene expression data. Appl Soft Comput 35:637–651. https://doi.org/10.1016/j.asoc.2015.06.019
Nepomuceno JA, Troncoso A, Nepomuceno-Chamorro IA, Aguilar-Ruiz JS (2016) Biclustering of gene expression data based on simUI semantic similarity measure. Hybrid Artif Intell Syst 9648:685–693. https://doi.org/10.1007/978-3-319-32034-2_57
Orzechowski P, Boryczko K (2016) Text mining with hybrid biclustering algorithms. In: Proceedings of the artificial intelligence and soft computing—15th international conference, (ICAISC'16), Part II, pp 102–113. https://doi.org/10.1007/978-3-319-39384-1_9
Orzechowski P, Sipper M, Huang X, Moore JH (2018) EBIC: an artificial intelligence-based parallel biclustering algorithm for pattern discovery. Comput Res Repos (CoRR) abs/1801.03039. arXiv:1801.03039
Padilha VA, Campello RJGB (2017) A systematic comparative evaluation of biclustering techniques. BMC Bioinform 18(1):55:1–55:25. https://doi.org/10.1186/s12859-017-1487-1
Pontes B, Divina F, Giráldez R, Aguilar-Ruiz JS (2007) Virtual error: A new measure for evolutionary biclustering. In: Evolutionary computation, machine learning and data mining in bioinformatics, 5th

European conference, EvoBIO 2007, Valencia, Spain, April 11–13, 2007, Proceedings, pp 217–226. https://doi.org/10.1007/978-3-540-71783-6_21

Pontes B, Giráldez R, Aguilar-Ruiz JS (2010) Measuring the quality of shifting and scaling patterns in biclusters. In: Pattern recognition in bioinformatics—5th IAPR international conference, PRIB 2010, Nijmegen, The Netherlands, September 22–24, 2010. Proceedings, pp 242–252. https://doi.org/10.1007/978-3-642-16001-1_21

Pontes B, Giráldez R, Aguilar-Ruiz JS (2013) Configurable pattern-based evolutionary biclustering of gene expression data. Algorithms Mol Biol 8:4. https://doi.org/10.1186/1748-7188-8-4

Pontes B, Giráldez R, Aguilar-Ruiz JS (2015) Biclustering on expression data: a review. J Biomed Inform 57:163–180. https://doi.org/10.1016/j.jbi.2015.06.028

Prelic A, Bleuler S, Zimmermann P, Wille A, Bühlmann P, Gruissem W, Hennig L, Thiele L, Zitzler E (2006) A systematic comparison and evaluation of biclustering methods for gene expression data. Bioinformatics 22(9):1122–1129. https://doi.org/10.1093/bioinformatics/btl060

Quackenbush J (2006) Microarray analysis and tumor classification. N Engl J Med 354(23):2463–2472. https://doi.org/10.1056/NEJMra042342

Robinson PN, Wollstein A, Böhme U, Beattie BJ (2004) Ontologizing gene-expression microarray data: characterizing clusters with gene ontology. Bioinformatics 20(6):979–981. https://doi.org/10.1093/bioinformatics/bth040

Rodríguez-Baena DS, Pérez-Pulido AJ, Aguilar-Ruiz JS (2011) A biclustering algorithm for extracting bit-patterns from binary datasets. Bioinformatics 27(19):2738–2745. https://doi.org/10.1093/bioinformatics/btr464

Saeed T, Jason HD, Michael CJ, Raymond CJ, George CM (1999) Systematic determination of genetic network architecture. Nat Genet 22(3):281–285

Sang C, Sun D (2014) Co-clustering over multiple dynamic data streams based on non-negative matrix factorization. Appl Intell 41(2):487–502. https://doi.org/10.1007/s10489-014-0526-0

Schaffer JD (1985) Multiple objective optimization with vector evaluated genetic algorithms. In: Proceedings of the 1st international conference on genetic algorithms, Pittsburgh, PA, USA, July 1985, pp 93–100

Seridi K, Jourdan L, Talbi E (2011) Multi-objective evolutionary algorithm for biclustering in microarrays data. In: Proceedings of the IEEE congress on evolutionary computation, CEC 2011, New Orleans, LA, USA, 5–8 June, 2011, pp 2593–2599. https://doi.org/10.1109/CEC.2011.5949941

Seridi K, Jourdan L, Talbi E (2015) Using multiobjective optimization for biclustering microarray data. Appl Soft Comput 33:239–249. https://doi.org/10.1016/j.asoc.2015.03.060

Serin A, Vingron M (2011) DeBi: discovering differentially expressed biclusters using a frequent itemset approach. Algorithms Mol Biol 6:18. https://doi.org/10.1186/1748-7188-6-18

Sharan R, Porat UB, Bleiberg O (2006) Analysis of biological networks: network modules—clustering and biclustering. Lecture 5:9

Shyama D, Mary IS (2010) Application of greedy randomized adaptive search procedure to the biclustering of gene expression data. Int J Comput Appl 2(3):6–13. https://doi.org/10.5120/650-907

Teng L, Chan L (2008) Discovering biclusters by iteratively sorting with weighted correlation coefficient in gene expression data. J Signal Process Syst 50(3):267–280. https://doi.org/10.1007/s11265-007-0121-2

Thangavel K, Bagyamani J, Rathipriya R (2012) Novel hybrid PSO-SA model for biclustering of expression data. Procedia Eng 30:1048–1055. https://doi.org/10.1016/j.proeng.2012.01.962

Trang T, Chi NC, Minh HN (2007) Management and analysis of DNA microarray data by using weighted trees. J Glob Optim 39(4):623–645. https://doi.org/10.1007/s10898-007-9158-9

Valente AF, Ayadi W, Elloumi M, Oliveira J, Oliveira J, Kao HJ (2013) A survey on biclustering of gene expression data. Biological knowledge discovery handbook: preprocessing, mining, and postprocessing of biological data. Wiley, Hoboken, pp 591–608. https://doi.org/10.1002/9781118617151.ch25

Wang J, Zaki MJ, Toivonen H, Shasha D (2005) Data mining in bioinformatics, advanced information and knowledge processing: chapter introduction to data mining in bioinformatics. Springer, Singapore, pp 3–8

Yang YH, Buckley MJ, Speed TP (2001) Analysis of CDNA microarray images. Brief Bioinform 2(4):341–349. https://doi.org/10.1093/bib/2.4.341

Yip K (2003) DB seminar series: biclustering methods for microarray data analysis, pp 46–47. http://www.cs.wayne.edu/shiyong/csc7710/assignments/biclusterppt

Yun T, Yi GS (2013) Biclustering for the comprehensive search of correlated gene expression patterns using clustered seed expansion. BMC Genom 14(1):144. https://doi.org/10.1186/1471-2164-14-144