Ben saber and Elloumi *BioData Mining* (2015) 8:38
DOI 10.1186/s13040-015-0070-4

BioData Mining

RESEARCH Open Access

A novel biclustering algorithm of binary microarray data: BiBinCons and BiBinAlter

Haifa Ben Saber[1*] and Mourad Elloumi[1,2]

*Correspondence:
bensaberhaifa1@gmail.com
[1]Latice laboratory, ENSIT, Tunis Time université, Tunis, Tunisia
Full list of author information is available at the end of the article

Abstract

The biclustering of microarray data has been the subject of a large research. No one of the existing biclustering algorithms is perfect. The construction of biologically significant groups of biclusters for large microarray data is still a problem that requires a continuous work. Biological validation of biclusters of microarray data is one of the most important open issues. So far, there are no general guidelines in the literature on how to validate biologically extracted biclusters. In this paper, we develop two biclustering algorithms of binary microarray data, adopting the *Iterative Row and Column Clustering Combination* (IRCCC) approach, called *BiBinCons* and *BiBinAlter*. However, the *BiBinAlter* algorithm is an improvement of *BiBinCons*. On the other hand, *BiBinAlter* differs from *BiBinCons* by the use of the *EvalStab* and *IndHomog* evaluation functions in addition to the *CroBin* one (Bioinformatics 20:1993–2003, 2004). *BiBinAlter* can extracts biclusters of good quality with better *p-values*.

Keywords: Biclustering, Algorithm, Evaluation function, Microarray data analysis

Introduction

DNA microarray technology is a revolutionary tool enabling the measurement of expression levels of thousands of genes in a single experiment under diverse experimental conditions. This technology allows us to obtain big raw data that can provide a wealth of information on the concerned genes. It proved to be a valuable tool for many biological and medical applications. Indeed, microarray data analysis is a crucial step for these applications in order to extract pertinent biological knowledge embedded in these large masses of data. However, the extraction process of this knowledge is far from being trivial. From here comes the necessity to adopt *data mining* techniques. Many of these techniques were applied to these data in order to extract pertinent biological knowledge. Among the techniques that are used, we mention those of *clustering* [1]. Indeed, by making a *clustering*, we consider that all the genes of a group can have a similar behavior under all the conditions. However, there are genes that have a similar behavior only under a subset of conditions. Hence, clustering is too simplistic to detect such cases [1]. Another more interesting technique, called *biclustering* [2], allows to identify groups of genes that have a similar behavior only under a subset of conditions.

In this paper, we develop new biclustering algorithms of microarray data. These data are usually coded by a data matrix $M(I,J)$, where the i^{th} row, $i \in I = \{1,2,\ldots,n\}$, represents the i^{th} gene, the j^{th} column, $j \in J = \{1,2,\ldots,m\}$, represents the j^{th} condition and the cell $M[i,j]$ represents the expression level of the i^{th} gene under the j^{th} condition.

The main objective is then to identify groups of genes that are coherent under groups of conditions, these groups are called *biclusters*. Genes belonging to the same bicluster have close biological functions. Let's note that, in its general form, the biclustering problem is NP-hard [2].

The rest of this chapter is organized as follow: In the second section, we introduce some preliminaries. In the third section, we present the *BiBinCons* algorithm. In the fourth section, we present the *BiBinAlter* algorithm. In the fifth section, we present an illustrative example and an experimental study. Finally, we present the conclusion of this paper.

Preliminaries

As we said in the introduction, the biclustering algorithms that we present in this paper are based on *CroBin* [1] function for the evaluation of a group of biclusters. So, let's present some preliminaries related to this function. Let $I = \{1, 2, \ldots, n\}$ be a set of indices of n genes, $J = \{1, 2, \ldots, m\}$ be a set of indices of m conditions and $M_b(I, J) = \left(m_{ij}^b\right)$, $i \in I$ and $j \in J$, be a binary data matrix associated with I and J. The biclustering problem of a binary microarray data can be formulated as a minimization of the criterion $W(z, w, a)$:

$$W(z, w, a) = \sum_{k=1}^{a} \sum_{l=1}^{m} \sum_{i \in z_k} \sum_{j \in w_l} \left| m_{ij}^b - a_{kl} \right|. \tag{2.1}$$

where

$z = \{z, z_2, \ldots, z_g\}$ is the matrix defined as a partition of I into g clusters, i.e. z_i is the cluster number of the i^{th} row of $M_b(I, J)$.

$w = \{w_1, w_2, \ldots, w_h\}$ is the matrix defined as a partition of J into h clusters, i.e. w_i is the cluster number of the j^{th} column of $M_b(I, J)$.

$a = (a_{kl})$ is a *summary matrix* of $M_b(I, J)$, it is a binary $g \times h$ matrix where k (resp. l) is the number of clusters on rows (resp. columns) and a_{kl} is defined by the m_{ij}'s satisfaying the following condition:

$$z_{ik} w_{jl} = 1 \tag{2.2}$$

where

$z_{ik} = 1$ if the i^{th} row of $M_b(I, J)$ belongs to the k^{th} cluster of I otherwise $z_{ik} = 0$.
$w_{jl} = 1$ if the j^{th} column of $M_b(I, J)$ belongs to the l^{th} cluster of J otherwise $w_{jl} = 0$.

By using Eq. (2.2), Eq. (2.1) can be reformulated as follows:

$$W(z, w, a) = \sum_{i,j,k,l} z_{ik} w_{jl} \left| m_{ij}^b - a_{kl} \right| \tag{2.3}$$

By adopting the IRCCC approach, we can make biclustering by minimizing $W(z, w, a)$ defined by Eq. (2.3) and by fixing either w or z:

- If w is fixed, the minimization is given by:

$$W(z, a|w) = \sum_{i,k,l} z_{ik} |u_{il} - (|w_l| \times a_{kl})| \tag{2.4}$$

where $u_{il} = \sum_{j \in w_l} m_{ij} = \sum_j w_{jl} m_{ij}$, $\sum_{i,j,k,l} z_{ik} w_{jl} \left| m^b_{ij} - a_{kl} \right| = \sum_{i,k} z_{ik} \sum_{j,l} w_{jl} \left| m^b_{ij} - a_{kl} \right| = \sum_{i,k} z_{ik} \sum_{l} |u_{il} - (|w_l| \times a_{kl})|$, u is a matrix of size $|I| \times l$.

- If z is fixed, the minimization is given by:

$$W(w, a|z) = \sum_{i,k,l} w_{jl} |v_{jl} - (|z_k| \times a_{kl})| \tag{2.5}$$

where $v_{kj} = \sum_{i \in z_k} m^b_{ij} = \sum_i z_{ik} m_{ij}$, $\sum_{i,j,k,l} z_{ik} w_{jl} \left| m^b_{ij} - a_{kl} \right| = \sum_{j,l} w_{jl} \sum_{i,k} z_{ik} \left| m^b_{ij} - a_{kl} \right| = \sum_{i,k} z_{ik} \sum_{l} |v_{kj} - (|z_k| \times a_{kl})|$, v is a matrix of size $k \times |J|$.

Remark. A colored block in the binary matrix $M_b(I, J)$ will be represented by a colored cell in the summary matrix A, where each colored cell contains the majority binary value in the corresponding colored block, e.g, if the majority of cells in a block in $M_b(I, J)$ contains 1 then the corresponding cell in A contains also 1.

Example. This example shows a binary data matrix $M_b(I, J)$ and the corresponding cell in the summary matrix A.

1	1	1	0	1	0
1	0	0	1	0	0
0	0	0	0	1	0
0	1	0	1	1	1
1	0	1	0	1	0
1	1	1	1	1	0
1	1	0	1	1	1
0	1	1	1	1	1
1	1	0	1	0	1
1	1	1	0	1	1
1	1	1	1	1	1

$M_b(I, J)$

0	0
1	1
1	1

Summary matrix A

$A = (0, 0; 1, 1; 1, 1)$, i.e., $a_{11} = 0, a_{12} = 0; a_{21} = 1, a_{22} = 1; a_{31} = 1, a_{32} = 1$.

In the section 'FIRST IRCCC Algorithm: *BiBinCons*', we develop two IRCCC algorithms of biclustering of binary microarray data, called respectively *BiBinCons* and *BiBinAlter*.

FIRST IRCCC Algorithm: *BiBinCons*

Our biclustering algorithm, *BiBinCons* receives as input a binary matrix $M_b(I, J)$ and gives as output $(z_{opt}, w_{opt}, A_{opt})$, where z_{opt} and w_{opt} are respectively the final clustering of rows and columns of $M_b(I, J)$, and A_{opt} is the summary matrix related to z_{opt} and w_{opt}. To describe more formally our biclustering algorithm, *BiBinCons*, we use the following notations:

z_0 : initial clustering of rows of $M_b(I, J)$

w_0 : initial clustering of columns of $M_b(I, J)$,

A_0 : initial summary matrix related to z^0 and w^0

z_c : current clustering of rows of $M_b(I, J)$

w_c : current clustering of columns of $M_b(I, J)$,

$A_c^{'}$: current intermidate summary matrix related to z^c and w^{c-1}

A_c : current summary matrix related to z^c and w^c
z_{opt} : final clustering of rows of $M_b(I,J)$
w_{opt} : final clustering of columns of $M_b(I,J)$
A_{opt} : final summary matrix related to z^{opt} and w^{opt}
$A_c^{'}$: intermediate current summary matrix.

Algorithm 1 *BiBinCons*

input : $M_b(I,J)$
output : $(z_{opt}, w_{opt}, A_{opt})$
Compute (z_0, w_0, A_0) thanks to the initialization step of *BiMax* algorithm Prelić et al. [3], // Initialization step :
$c := 1$
while $(z_c, w_{c-1}, A_c^{'}) \neq (z_{c-1}, w_{c-1}, A_{c-1})$ **do** // Clustering of rows
Compute $(z_c, w_{c-1}, A_c^{'})$ starting from $(z_{c-1}, w_{c-1}, A_{c-1})$, by using Eq. 2.4
end while
$(z_{c-1}, w_{c-1}, A_{c-1}) := (z_c, w_{c-1}, A_c^{'})$
while$(z_c, w_c, A_c)) \neq (z_{c-1}, w_{c-1}, A_{c-1})$ **do** // Clustering of columns
Compute (z_c, w_c, A_c) starting from $(z_c, w_{c-1}, A_c^{'})$, by using Eq. 2.5
end while
$(z_{opt}, w_{opt}, A_{opt}) := (z^c, w^c, A^c)$
return $(z_{opt}, w_{opt}, A_{opt})$

Second IRCCC Algorithm: *BiBinAlter*

Our biclustering algorithm, *BiBinAlter* receives as input a binary matrix $M_b(I,J)$ and gives as output $(z_{opt}, w_{opt}, A_{opt})$, where z_{opt} and w_{opt} are respectively the final clustering of rows and columns of $M_b(I,J)$, and A^{opt} is the summary matrix related to z_{opt} and w_{opt}. By adopting *BiBinAlter*, we propose the use of functions defined:

$EvalStab_c$ represents the frequency of 0's in the current group of biclusters at the c^{th} iteration. It is defined as follows:

$$EvalStab = \sum_{k,l} \frac{|a_{kl} - (|z_k| \times |w_l|)|}{|z_k||w_l|} \tag{4.1}$$

$IndHomog_c$ represents the tradeoff between the number of mixed biclusters (containing both 0's and 1's) and the total number of biclusters at the c^{th} iteration. It is defined as follows:

$$IndHomog = \frac{MixedBic}{AllBic} \tag{4.2}$$

To describe more formally our biclustering algorithm, *iBinAlter*, we have used the same notations like previous algorithm besides of these notations:

$(EvalStab_c, IndHomog_c)$: couple to present the frequency of 0's in the current group of biclusters at the c^{th} iteration and the tradeoff between the number of mixed biclusters (containing both 0's and 1's) and the total number of biclusters at the c^{th} iteration.

$(EvalStab_{c-1}, IndHomog_{c-1})$: couple to present the frequency of 0's in the group of biclusters at the $(c-1)^{th}$ iteration and the tradeoff between the number of mixed biclusters (containing both 0's and 1's) and the total number of biclusters at the $(c-1)^{th}$ iteration.

$\left(EvalStab'_{(c-1)}, IndHomog'_{(c-1)}\right)$: couple to present the frequency of 0's in the group of biclusters at the intermidate $(c-1)^{'th}$ iteration and the tradeoff between the number of mixed biclusters (containing both 0's and 1's) and the total number of biclusters at the intermidate $(c-1)^{'th}$ iteration.

Algorithm 2 *BiBinAlter*

input : $M_b(I,J)$
output : $(z_{opt}, w_{opt}, A_{opt})$
// Initialization step :
compute (z_0, w_0, A_0)

// Biclustering step :
while $(((z_c, w_c, A_c) \neq (z_{c-1}, w_{c-1}, A_{c-1}))$ **and** $(EvalStab_c, IndHomog_c)$ $\neq (EvalStab_{c-1}, IndHomog_{c-1})))$ **or** $(((z_c, w_c, A_c) \neq (z_c, w_{c-1}, A'_c))$ **and** $((EvalStab_c, IndHomog_c)$ $\neq (EvalStab'_{(c-1)}, IndHomog'_{(c-1)}))$ **do**
Compute(z_c, w_{c-1}, A'_c) starting from $(z_{c-1}, w_{c-1}, A_{c-1})$, by using Eq. 2.4
//A' is an intermediate summary matrix ;
Compute $(EvalStab'_c, IndHomog'_c)$, by using Eqs. 4.1 and 4.2
Compute (z_c, w_c, A_c) starting from (z_c, w_{c-1}, A'_c), by using Eq. 2.5
Compute $(EvalStab_c, IndHomog_c)$, by using Eqs. 4.1 and 4.2
end while
$(z_{opt}, w_{opt}, A_{opt}) := (z_c, w_c, A_c)$
return $(z_{opt}, w_{opt}, A_{opt})$

Illustrative example

Let's apply the *BiBinAlter* algorithm on the following binary matrix $M_b(I,J)$:

	C_1	C_2	C_3	C_4	C_5
G_1	1	1	0	1	0
G_2	0	0	1	0	1
G_3	1	1	0	1	0
G_4	0	0	1	0	1

$M_b(I,J)$

- **Initialization step**

First, we initialize the rows and columns thanks to the initialization step of *BiMax* algorithm Prelić [3] and we compute (z_0, w_0, A_0), we obtain:

$$z_0 = (1,2,2,3),\ w_0 = (1,1,0,0,0),\ A_0 = (1,0;1,1;0,1)$$

	C_1	C_2	C_3	C_4	C_5	Z_0
G_1	1	1	0	1	0	1
G_2	0	0	1	0	1	2
G_3	1	1	0	1	0	2
G_4	0	0	1	0	1	3
W_0	1	1	2	2	2	

$M_b(I,J)$

1	0
1	1
0	1

A_0

A colored block in the binary matrix $M_b(I,J)$ will be represented by a colored cell in the summary matrix A_0, where each colored cell contains the majority binary value in the corresponding colored block, e.g, if the majority of cells in a block in $M_b(I,J)$ contains 1 then the corresponding cell in A_0 contains also 1.

- **Biclustering step:**

Iteration 1: $c = 1$

We compute $(z_1, w_0, A_1^{'})$ starting from (z_0, w_0, A_0) by using Eq. 2.4, we obtain:

$$\left(z_1, w_0, A_1^{'}\right) = ((1,3,2,1),(1,1,2,2,2),(1,1;1,0;0,1))$$

We compute $(EvalStab_1^{'}, IndHomog_1^{'})$ by using Eq. 4.3, we obtain:

$$\left(EvalStab_1^{'}, IndHomog_1^{'}\right) = \left(2, \frac{2}{3}\right)$$

We compute (z_1, w_1, A_1) starting from $(z_1, w_0, A_1^{'})$, by using Eq. 2.4, we obtain:

$$(z_1, w_1, A_1) = ((1,3,2,1),(2,2,1,2,1),(1,1;1,0;0,1))$$

We compute $(EvalStab_1, IndHomog_1)$, by using Eq. 4.3, we obtain:

$$\left(EvalStab_1, IndHomog_1\right) = \left(1, \frac{2}{6}\right)$$

Since we have

$$(((z_c, w_c, A_c) \neq (z_{c-1}, w_{c-1}, A_{c-1})) \textbf{ and } (EvalStab_c, IndHomog_c) \neq (EvalStab_{c-1}, IndHomog_{c-1}))$$

and

$$(((z_c, w_c, A_c) \neq (z_c, w_{c-1}, A_c^{'})) \textbf{ and } ((EvalStab_c, IndHomog_c) \neq (EvalStab_{(c-1)}^{'}, IndHomog_{(c-1)}^{'}))$$

we make another iteration

Iteration 2: $c = 2$

We compute $(z_2, w_1, A_2^{'})$ starting from (z_1, w_1, A_1), by using Eq. 2.4, we obtain:

$$\left(z_2, w_1, A_2^{'}\right) = ((2,1,2,3),(2,2,1,2,1),(0,1;1,0,0,1))$$

	C_1	C_2	C_3	C_4	C_5	Z_2
G_1	1	1	0	1	0	2
G_2	0	0	1	0	1	1
G_3	1	1	0	1	0	2
G_4	0	0	1	0	1	3
W_1	2	2	1	2	1	

$M_b(I,J)$

0	1
1	0
0	1

$A_2^{'}$

We compute $(EvalStab_2^{'}, IndHomog_2^{'})$, by using Eq. 4.3, we obtain:

$$\left(EvalStab_2^{'}, IndHomog_2^{'}\right) = \left(0, \frac{0}{6}\right)$$

We compute (z_2, w_2, A_2) starting from $(z_2, w_1, A_2^{'})$, by using Eq. 2.4, we obtain:

$$(z_2, w_2, A_2) = ((2,1,2,3),(2,2,1,2,1),(0,1;1,0;0,1))$$

We compute $(EvalStab^2, IndHomog^2)$, by using Eq. 4.3, we obtain:

$$\left(EvalStab^2, IndHomog^2\right) = \left(0, \frac{0}{6}\right)$$

Since we have

$$(((z_c, w_c, A_c) \neq (z_{c-1}, w_{c-1}, A_{c-1})) \textbf{ and } (EvalStab_c, IndHomog_c) \neq (EvalStab_{c-1}, IndHomog_{c-1})))$$

and

$$(((z_c, w_c, A_c) \neq (z_c, w_{c-1}, A_c^{'})) \textbf{ and } ((EvalStab_c, IndHomog_c) = (EvalStab_{(c-1)}^{'}, IndHomog_{(c-1)}^{'}))$$

we make another iteration

Iteration 3: $c = 3$

We compute $(z_3, w_2, A_3^{'})$ starting from (z_2, w_2, A_2), by using Eq. 2.4, we obtain:

$$\left(z_3, w_2, A_3^{'}\right) = ((1,1,1,1),(2,2,1,2,1),(1,1)) \tag{4.3}$$

	C_1	C_2	C_3	C_4	C_5	Z_3
G_1	1	1	0	1	0	1
G_2	0	0	1	0	1	1
G_3	1	1	0	1	0	1
G_4	0	0	1	0	1	1
W_2	2	2	1	2	1	

$M_b(I,J)$

1	1

$A_3^{'}$

We compute $(EvalStab_3^{'}, IndHomog_3^{'})$, by using Eq. 4.3, we obtain:

$$\left(EvalStab_3^{'}, IndHomog_3^{'}\right) = (1,1)$$

We compute (z_3, w_3, A_3) starting from $(z_3, w_2, A_3^{'})$, by using Eq. 2.4, we obtain:

$$(z_3, w_3, A_3) = ((2,1,2,3),(2,2,1,2,1),(0,1;1,0;0,1))$$

We compute $(EvalStab^3, IndHomog^3)$, by using Eq. 4.3, we obtain:

$$\left(EvalStab^3, IndHomog^3\right) = \left(0, \frac{0}{6}\right)$$

Since we have

$$(((z_c, w_c, A_c) \neq (z_{c-1}, w_{c-1}, A_{c-1})) \textbf{ and } (EvalStab_c, IndHomog_c) \neq (EvalStab_{c-1}, IndHomog_{c-1})))$$

and

$$(((z_c, w_c, A_c) \neq (z_c, w_{c-1}, A_c^{'})) \textbf{ and } ((EvalStab_c, IndHomog_c) \neq (EvalStab_{(c-1)}^{'}, IndHomog_{(c-1)}^{'}))$$

we make another iteration

Iteration 4: $c = 4$

We compute $(z_4, w_3, A_4^{'})$ starting from (z_3, w_3, A_3), by using Eq. 2.4, we obtain:

$$\left(z_4, w_3, A_4^{'}\right) = ((1,2,1,2),(2,2,1,2,1),(1,0;0,1))$$

	C_1	C_2	C_3	C_4	C_5	Z_4
G_1	1	1	0	1	0	1
G_2	0	0	1	0	1	2
G_3	1	1	0	1	0	1
G_4	0	0	1	0	1	2
W_3	2	2	1	2	1	

$M_b(I,J)$

1	0
0	1

$A_4^{'}$

We compute $\left(EvalStab_4^{'}, IndHomog_4^{'}\right)$, by using Eq. 4.3, we obtain:

$$\left(EvalStab_4^{'}, IndHomog_4^{'}\right) = \left(0, \frac{0}{4}\right)$$

We compute (z_4, w_4, A_4) starting from $(z_4, w_3, A_4^{'})$, by using Eq. 2.4, we obtain:

$$(z_4, w_4, A_4) = ((1,2,1,2),(1,1,1,1,1),(1;0))$$

We compute $\left(EvalStab^4, IndHomog^4\right)$, by using Eq. 4.3, we obtain:

$$\left(EvalStab^4, IndHomog^4\right) = (1,1)$$

Since we have

$$(((z_c, w_c, A_c) \neq (z_{c-1}, w_{c-1}, A_{c-1})) \textbf{ and } (EvalStab_c, IndHomog_c) \neq (EvalStab_{c-1}, IndHomog_{c-1})))$$

and

$$(((z_c, w_c, A_c) \neq (z_c, w_{c-1}, A_c^{'})) \textbf{ and } ((EvalStab_c, IndHomog_c) \neq (EvalStab_{(c-1)}^{'}, IndHomog_{(c-1)}^{'}))$$

we make another iteration

Iteration 5: $c = 5$

We compute $(z_5, w_4, A_5^{'})$ starting from (z_4, w_4, A_4), by using Eq. 2.4, we obtain:

$$\left(z_5, w_4, A_5^{'}\right) = ((1,2,1,2),(1,1,1,1,1),(1;0))$$

	C_1	C_2	C_3	C_4	C_5	Z_5
G_1	1	1	0	1	0	1
G_2	0	0	1	0	1	2
G_3	1	1	0	1	0	1
G_4	0	0	1	0	1	2
W_4	1	1	1	1	1	

$M_b(I,J)$

1
0

$A_5^{'}$

We compute $(EvalStab_5^{'}, IndHomog_5^{'})$, by using Eq. 4.3, we obtain:

$$\left(EvalStab_5^{'}, IndHomog_5^{'}\right) = (1,1)$$

We compute (z_5, w_5, A_5) starting from $(z_5, w_4, A_5^{'})$, by using Eq. 2.4, we obtain:

$$(z_5, w_5, A_5) = ((1,2,1,2),(1,1,2,1,2),(1,0;0,1))$$

We compute $(EvalStab^5, IndHomog^5)$, by using Eq. 4.3, we obtain:

$$\left(EvalStab^5, IndHomog^5\right) = \left(0, \frac{0}{4}\right)$$

We have $(EvalStab_4^{'}, IndHomog_4^{'}) = (EvalStab^5, IndHomog^5))$ and $(z_5, w_5, A_5) = (z_4, w_3, A_4^{'})$.

Since we have

$$(((z_c, w_c, A_c) = (z_c, w_{c-1}, A_c^{'})) \textbf{ and } ((EvalStab_c, IndHomog_c) = (EvalStab_{(c-1)}^{'}, IndHomog_{(c-1)}^{'}))$$

we stop the loop.

Then, we obtain $(z_{opt}, w_{opt}, A_{opt}) = (z_5, w_5, A_5)$. Biclusters that contain only 0's will not be considered because they represent genes that are not expressed under the related conditions. Finally, $(z_{opt}, w_{opt}, A_{opt})$ can be represented in $M_b(I,J)$ as follows:

	C_1	C_2	C_3	C_4	C_5	Z_5
G_1	1	1	0	1	0	1
G_2	0	0	1	0	1	2
G_3	1	1	0	1	0	1
G_4	0	0	1	0	1	2
W_5	1	1	2	1	2	

$M_b(I,J)$

Results for synthetic datasets

In this section, we present an experimental study to evaluate the performance of our algorithms of microarray data. Indeed, we compare the results of our algorithms to those

obtained by a selection of known algorithms cited in the literature. We conducted experiments on synthetic and real datasets of microarrays. The idea behind testing on synthetic datasets is to investigate the ability of our algorithms to extract different types of biclusters. However, on real datasets, we seek to assess the degree of response of our algorithms for statistical and biological criteria.

Synthetic microarray datasets and comparaison criteria

By adopting the strategy and data described in [1], we have experimented our algorithms on synthetic datasets by operarting as follows: First, we choose the number of biclusters, 3 clusters on rows ($g = 3$) and 2 clusters on columns ($m = 2$). Second, we use the *Latent Bernoulli Mixture* (LBM) model [1] to generate binary matrices (*mixtures*) by considering:

(*a*) Overlapping biclusters (overlapping rate = 5 % (well separated), 15 % (fairly separated) and 25 % (poorly separated)).
(*b*) Different data sizes (matrix size = 50×30 (small), 100×60 (medium) and 200×120 (large)).

Similar to [4], we use two indices, *Recovery* and *Relevance*, to evaluate our biclustering algorithms: Let B_1 be a group of true implemented biclusters in a binary data matrix M_b and B_2 be a group of output biclusters of a biclustering algorithm, *Relevance* reflects to what extent B_2 is similar to B_1, while *Recovery* quantifies how well each bicluster in B_1 is recovered by B_2 [3]:

$$Recovery = Overlap(B_2, B_1) \tag{6.1}$$

$$Relevance = Overlap(B_1, B_2) \tag{6.2}$$

where:

$$Overlap(B_1, B_2) = \frac{1}{|B1|} \sum_{(I_1,J_1)\in B_1} \max_{(I_2,J_2)\in B_2} \frac{|I_1 \cap I_2||J_1 \cap J_2|}{|I_1 \cup I_2||J_1 \cup J_2|} \tag{6.3}$$

We use also two other indices cited in [2],

$$Shared = \frac{S_{cb}}{Tot_{size}} 100 \tag{6.4}$$

$$NotShared = \frac{S_{ncb}}{Tot_{size}} 100 \tag{6.5}$$

where

S_{cb} is the volume of correctly extracted biclusters, Tot_{size} is the total volume of implemented biclusters and S_{NCB} is the volume of not correctly extracted biclusters.

The *Shared* index (resp. *NotShared*) represents the percentage of correctly (resp. not correctly) extracted biclusters with respect to all implemented biclusters in the data matrix. Indeed, when the *Shared* value is equal to 100 %, the algorithm extracts all the implemented biclusters. When the value of *NotShared* is 0 %, the algorithm extracts no cell outside the implemented biclusters.

Table 1 Corresponding parameters values of our algorithms

Algorithms	Corresponding parameters values
BiBinCons	*minrow* = 2, *mincol* = 2
BiBinAlter	*minrow* = 2, *mincol* = 2

Table 2 Values of *Shared* and *NotShared* for non overlapping biclusters

Algorithms	*Shared*	*NotShared*
CC	18.21 %	36.57 %
OPSM	46.39 %	74.42 %
ISA	39.38 %	5.31 %
BiMax	58.18 %	21.39 %
BiBinCons	88 %	12 %
BiBinAlter	100 %	37.03 %

Experimental protocol

We have compared our algorithms to CC Cheng and Wee-Chung [2], OPSM Ben-Dor and Yakhini [5], ISA Ihmels et al. [4] and BiMax Kaiser and Leisch [6]. These algorithms were implemented in the *BIClustering Analysis Toolbox* (Bicat) platform. After several simulations, the parameters of our algorithms were set as listed in Table 1. Indeed, at each simulation, we set a parameter and we vary the other and vice inverse. Finally, we keep the parameters which give the nearest implemented biclusters in the starting template.

For CC, OPSM, ISA and BiMax algorithms, we keep the value of the default parameters values. Indeed, these values give biclusters of reasonable quality. We have adopted *Shared*, *NotShared*, *Recovery* and *Relevance* as comparaison criteria. Table 2 shows the best biclusters extracted by each algorithm:

As we can notice in Table 2, for the generated binary matrices, the best values of *Shared* and *NotShared* for non overlapping biclusters were obtained by the *BiBinAlter* algorithm. Indeed, to get a solution B_{opt}, the combination between two biclusters provides additional volume for the conditions. This reasonnable additional volume is generated by a successive comparaisons between P_{max} and the other polynoms of L, and we locate the polynom $P_{uncomon}$ that has the lowest number ρ of uncommon terms with P_{max}. In fact, the interesting resulat is obtained because of we keep only the conditions that have not been removed by the pretreatment process. Besides, the extracted bicluster from the current matrix $M_b(I,J)$ is removed ans we set the cells of $M_b(I,J)$, representing the new bicluster, to 0. Table 3 shows the best biclusters extracted by each algorithm. As we can notice in Table 3, for the generated binary matrices, the best values of *Shared* and *NotShared* for overlapping biclusters were also obtained by the *BiBinAlter* algorithm. We can explain this as follows: *BiBinAlter* results covers most of the implemented biclusters. Table 4 presents the number of biclusters obtained bu our algorithms on real datastes.

Results of our algorithms on real datasets

In this section, we evaluate our algorithms on real microarray datasets.

Table 3 Values of *Shared* and *NotShared* for overlapping biclusters

Algorithms	*Shared*	*NotShared*
CC	13.21 %	36.57 %
OPSM	82.02 %	50.51 %
ISA	29.28 %	7.31 %
BiMax	48.18 %	22.39 %
BiBinCons	87.30 %	61 %
BiBinAlter	89.40 %	57.32 %

Table 4 Number of biclusters obtained by our algorithms on real datasets

Algorithms	*Yeast cell cycle*	*Human B-cell Lymphoma*
EnumLat	883	1921
DecBinBicluster	708	1720
BiBinCons	529	1900
BiBinAlter	881	1769
RefineBicluster	708	1700

Real microarray datasets

We have used two real microarray datasets: The Yeast cell cycle dataset which has been described and then pretreated in [1]. It contains the expression of 2884 genes in 17 terms ans the Human B-cell Lymphoma dataset which has been described by Alizadeh et al. [1], it contains 4026 genes and 96 conditions. These datasets are used frequently in the literature by biclustering algorithms.

Experimental protocol

The first experiments concern the statistical validation. It enables to calculate the coverage for *Yeast cell cycle* and *Human B-cell Lymphoma* datasets and the *p*-value adjusted for*Human B-cell Lymphoma* datasets. The second experiments was applied to *Yeast cell cycle* in order to study the biological significance of extracted biclusters.

Statistical validation

In order to validate statistically our algorithms on these real datasets, we evaluate the performance of *BiBinCons* and *BibinAlter*. We calculate the total number of cells covered by the biclusters. To do this, we have processed as in [2], and we have compared the results of our algorithms to those reported in [2]. In the literature, the coverage test was performed on *Yeast cell cycle* and *Human B-cell Lymphoma* datasets. This test is not applied to *RefineBicluster* algorithm because it is only a refinement algorithm.

Table 5 reports the percentage of *Coverage* on the different algorithms for *Yeast cell cycle* and *Human B-cell Lymphoma* datasets. We note that most algorithms have more or less close rates. For example, for the *Yeast cell cycle* datase, *BiBinCons* has the lowest performance. This is explained by the fact that *BiBinCons* extracts thousands of small sized biclusters. The CC algorithm extracts biclusters with random values. Thus, CC prohibits the genes/conditions already discovered to be selected in the next search process. This type of mask leads to a high coverage and preventing the discovery of large biclusters.

Biological validation

To evaluate biologically extracted biclusters, we use the web tool *GOTermFinder*. To do this, we present the most significant shared biclusters. In this section, we evaluate

Table 5 Values of *Coverage* for *Yeast cell cycle* and *Human B-cell Lymphoma* datasets

Datasets	Algorithms	Total coverage	Genes coverage	Conditions coverage
Yeast celll cycle	CC	81.47 %	97.12 %	100 %
	BiBinCons	39.14 %	44.5 %	100 %
	BiBinAlter	47 %	48,03 %	100 %
Human B-cell Lymphoma	CC	36.81 %	91.58 %	100 %
	BiBinCons	34.14 %	37.51 %	100 %
	BiBinAlter	41 %	46.13 %	100 %

Table 6 The most important terms of GO for the two most significant extracted biclusters from *Yeast cell cycle* dataset by *BiBinCons* and *BiBinAlter*

Biclusters	Biological process	Molecular function	Cell component
12 genes, 13 conditions	Cellular response to chromatin binding microtubule organizing 13 conditions DNA damage stimulus (25 %,0.00037) center part (66.7 %, 1:87 * 10-8) (16.7 %, 0.00742) response to DNA damage stimulus (66.7 %, 6:30 * 10-8) cellular response to stress (66.7 %, 2:12 * 10-7) cellular response to stimulus (66,7 %, 3:25 * 10-7) DNA repair (50 %, 2:58 * 10-5) response to stress (66.7 %, 2:98 * 10-5)	Chromatin binding microtubule organizing 13 conditions DNA damage stimulus (25 %,0.00037)	Microtubule organizing 13 conditions DNA damage stimulus (25 %,0.00037) center part (66.7 %, 1:87 * 10-8) (16.7 %, 0.00742)
11 genes, 11 conditions	Cell cycle process GTPase activator microtubule cytoskeleton 11 conditions (63.6 %, 2:93 * 10-5) activity (18.2 %,0.00994) (45.5 %, 6:33 * 10-6) cell cycle microtubule organizing (63.6 %, 6:85 * 10-5)	GTPase activator microtubule cytoskeleton 11 conditions (63.6 %, 2:93 * 10-5) activity (18.2 %,0.00994)	Microtubule cytoskeleton 11 conditions (63.6 %, 2:93 * 10-5) activity (18.2 %,0.00994) (45.5 %, 6:33 * 10-6) cell cycle microtubule organizing (63.6 %, 6:85 * 10-5) center (36.4 %,4:97 * 10-5) spindle pole body (36.4 %, 4:97 * 10-5) spindle pole (36.4 %, 6:77 * 10-5)

Table 7 Computing time of our algorithms

Datasets	*BiBinCons*	*BiBinAlter*
Yeast Cell Cycle	32 min	37 min 12 sec
Saccharomyces Cerevisiae	8 min	8 min 3 sec

BiBinCons and *BiBinAlter* algorithms on real microarray datasets. We have choosen this algorithm because it gaves the best results on synthetic datasets. Table 6 presents the most important terms of GO for the two most significant extracted biclusters from *Yeast cell cycle* dataset by *BiBinCons* and *BiBinAlter*.

Computing time

Table 7 shows the computing time of *BiBinCons* and *BiBinAlter* algorithms. All developed algorithms in this thesis were implemented in R under the R studio. The physical characteristics of the machine are as follows: a PC with an Intel Core 2 Duo T6400 with a clock frequency of 2.0 GHz and 3.5 GO of RAM. We note that *BiBinAlter* algorithm is the most time consuming and this is due to the use of proposed evaluation function.

Conclusion

In this paper, we have developed two biclustering algorithms of binary microarray data, called *BiBinCons* and *BiBinAlter*, adopting the *Iterative Row and Column Clustering Combination* (IRCCC) approach, however, the *BiBinAlter* algorithm is an improvement of *BiBinCons*. On the other hand, *BiBinAlter* differs from *BiBinCons* by the use of the *EvalStab* and *IndHomog* evaluation functions in addition to the *CroBin* one [1]. *BiBinAlter* can extract biclusters of good quality with better *p-values*. In this paper, we have presented an experimental study of our biclustering algorithms of microarray data. We have compared the results of our algorithms to those obtained by a selection of the known biclustering algorithms. We have conducted experiments on both synthetic and real datasets of microarrays. For both synthetic and real datasets, our biclustering algorithm *BiBinAlter* outperforms the other algorithms, followed by our other biclustering algorithms nd *BiBinCons*.

Competing interests
The authors declare that they have no competing interests.

Author details
[1] Latice laboratory, ENSIT, Tunis Time université, Tunis, Tunisia. [2] Latice laboratory, Ensit, Tunis Université tunis el manar, Tunis, Tunisia.

Received: 4 January 2015 Accepted: 8 November 2015
Published online: 30 November 2015

References

1. Govaert G. La classification croisee. Modulad. 1983.
2. Law NF, Siu WC, Cheng KO, Alan WC. Identification of coherent patterns in gene expression data using an efficient biclustering algorithm and parallel coordinate visualization. BMC Bioinformatics. 2008.
3. Prelić A, Bleuler S, Zimmermann P, Wille A, Bühlmann P, Gruissem W, et al. A systematic comparison and evaluation of biclustering methods for gene expression data. Bioinformatics. 2006;22:1122–29.
4. Ihmels J, Bergmann S, Barkai N. Defining transcription modules using large-scale gene expression data. Bioinformatics. 2004;20(13):1993–2003.
5. Benny C, Richard K, Amir BD, Yakhini Z. Discovering local structure in gene expression data: The order-preserving submatrix problem. In: Proceedings of the Sixth Annual International Conference on Computational Biology, RECOMB '02. New York, NY, USA: ACM; 2002. p. 49–57.
6. Santamaria R, Khamiakova T, Sill M, Theron R, Quintales L, Kaiser S, et al. biclust: Bicluster algorithms. R package. 2011.