BIOINFORMATICS ORIGINAL PAPER Vol. 26 no. 20 2010, pages 2594–2600
doi:10.1093/bioinformatics/btq473

Gene expression

Advance Access publication August 23, 2010

Improving performances of suboptimal greedy iterative biclustering heuristics via localization

Cesim Erten[1,*] and Melih Sözdinler[2,*]

[1]Department of Computer Engineering, Kadir Has University, Cibali, Istanbul 34083 and [2]Department of Computer Engineering, Bogaziçi University, Bebek, Istanbul 34342, Turkey

Associate Editor: Trey Ideker

ABSTRACT

Motivation: Biclustering gene expression data is the problem of extracting submatrices of genes and conditions exhibiting significant correlation across both the rows and the columns of a data matrix of expression values. Even the simplest versions of the problem are computationally hard. Most of the proposed solutions therefore employ greedy iterative heuristics that locally optimize a suitably assigned scoring function.

Methods: We provide a fast and simple pre-processing algorithm called *localization* that reorders the rows and columns of the input data matrix in such a way as to group correlated entries in small local neighborhoods within the matrix. The proposed localization algorithm takes its roots from effective use of graph-theoretical methods applied to problems exhibiting a similar structure to that of biclustering. In order to evaluate the effectivenesss of the localization pre-processing algorithm, we focus on three representative greedy iterative heuristic methods. We show how the localization pre-processing can be incorporated into each representative algorithm to improve biclustering performance. Furthermore, we propose a simple biclustering algorithm, *Random Extraction After Localization (REAL)* that randomly extracts submatrices from the localization pre-processed data matrix, eliminates those with low similarity scores, and provides the rest as correlated structures representing biclusters.

Results: We compare the proposed localization pre-processing with another pre-processing alternative, non-negative matrix factorization. We show that our fast and simple localization procedure provides similar or even better results than the computationally heavy matrix factorization pre-processing with regards to *H*-value tests. We next demonstrate that the performances of the three representative greedy iterative heuristic methods improve with localization pre-processing when biological correlations in the form of functional enrichment and PPI verification constitute the main performance criteria. The fact that the random extraction method based on localization REAL performs better than the representative greedy heuristic methods under same criteria also confirms the effectiveness of the suggested pre-processing method.

Availability: Supplementary material including code implementations in LEDA C++ library, experimental data, and the results are available at http://code.google.com/p/biclustering/

Contacts: cesim@khas.edu.tr; melihsozdinler@boun.edu.tr

Supplementary information: Supplementary data are available at *Bioinformatics* online.

*To whom correspondence should be addressed.

Received on April 13, 2010; revised on June 24, 2010; accepted on August 13, 2010

1 INTRODUCTION

Clustering refers to the process of organizing a set of input vectors into clusters based on similarity specified according to some predefined distance measure. In many cases, it is more desirable to simultaneously cluster the dimensions as well as the vectors themselves. This special instance of clustering, referred to as *biclustering*, was introduced by Hartigan (Hartigan, 1972). Although traditional one-way clustering provides valuable information with regards to a global perspective, extracting local substructures via biclustering may help build intuition on both dimensions of the data. In addition to the areas such as data mining and pattern recognition, biclustering has found many applications in bioinformatics, specifically in microarray analysis, drug activity analysis and motif detection (Barkow *et al.*, 2006; Ben-Dor *et al.*, 2002; Kluger *et al.*, 2003; Murali and Kasif, 2003; Prelić *et al.*, 2006; Tanay *et al.*, 2002).

In gene expression analysis, data are assumed to be arranged in a matrix, where each gene corresponds to a row and each condition to a column. After reordering rows and columns of the matrix, a bicluster can then be defined as a submatrix with significant correlation among its data values. Such a submatrix is likely to group together genes that exhibit similar behavior over a subset of experimental conditions.

One of the early approaches for biclustering expression data is that of Cheng and Church (Cheng and Church, 2000). A *mean-squared residue* score is defined and the algorithm greedily inserts/removes rows and columns to arrive at a certain number of biclusters achieving some predefined residue score. Order preserving submatrix (Ben-Dor *et al.*, 2002) is another greedy, iterative algorithm that finds a statistically significant bicluster at each iteration. Maximum similarity biclusters (MSB; Liu and Wang, 2007) starts by constructing a similarity matrix based on a reference gene. A greedy strategy of removing rows/columns iteratively is employed to provide the maximum similarity bicluster in polynomial time. Large average submatrices (LAS; Shabalin *et al.*, 2009) is a recently proposed algorithm which tries to extract large average submatrices according to a *Bonferroni-based* significance score. Several graph-theoretical approaches have also been suggested. Prelic *et al*. provided a divide-and-conquer algorithm, Bimax (Prelić *et al.*, 2006), that runs on discretized binary data. In SAMBA (Sharan *et al.*, 2003; Tanay *et al.*, 2002), the data matrix is viewed as a bipartite graph where the genes/conditions constitute the layers of the bipartite graph and edges in the graph correspond to the

expression changes. The goal is to find out heavy bicliques inside the graph. A similar model is constructed in (Abdullah and Hussain, 2006) where crossing minimization in unit-weight bipartite graphs is used as a means to extract bicliques corresponding to biclusters in the data matrix. The model is generalized to weighted bipartite graphs in (Erten and Sözdinler, 2009). Many other biclustering algorithms including xMOTIFs (Murali and Kasif, 2003), ISA (Bergmann *et al.*, 2003), coupled two-way clustering (Getz *et al.*, 2000) have also been suggested; see the survey of Madeira *et al*. for further details of various other biclustering methods (Madeira and Oliveira, 2004).

Once a model is determined, be it a matrix of real values corresponding to relative abundance of mRNA, a discretized binary matrix or a bipartite graph model of the data matrix, the problem becomes that of globally optimizing a suitable scoring function defined under the terms including *residue* (Cheng and Church, 2000), *similarity* (Liu and Wang, 2007), *order-preserving submatrix* (Ben-Dor *et al.*, 2002), *maximal biclique* (Tanay *et al.*, 2002). NP-hardness of even the simplest versions of all these seemingly similar optimization problems makes the task quite challenging (Alexe *et al.*, 2004; Madeira and Oliveira, 2004). An approach common to most of the existing algorithms, therefore, is to apply a greedy iterative heuristic that aims at locally improving the suitable scoring function under the defined model.

Following this observation and the common observation that a 'good' initial configuration is especially important in the success of greedy iterative heuristics for solving optimization problems (Srinivas and Patnaik, 1994), we present a pre-processing method called *localization* that provides an appropriate initial configuration by placing rows/columns exhibiting similar patterns in nearby locations within the data matrix. Although the majority of the proposed solutions for biclustering consists of iterative, greedy local optimization heuristics, no suitable pre-processing algorithm has been suggested for an improvement of these methods. We extract only one such method, non-singular non-negative matrix factorization (nsNMF) proposed by Carmona-Saez *et al.* (2006). Though not presented originally as a pre-processing method *per se*, but rather as a biclustering method in itself, we employ nsNMF for our comparisons as a possible alternative pre-processing method. This is plausible since reordering rows and columns so as to gather similar entities in close proximity is also a common goal of nsNMF. We show experimentally that our proposed pre-processing method of localization provides similar or even better results than the computationally heavy nsNMF. In addition, we show that when pre-processed with our localization method, performances of the greedy iterative biclustering heuristics improve. The algorithms under consideration are SAMBA (Tanay *et al.*, 2002), MSB (Liu and Wang, 2007) and LAS (Shabalin *et al.*, 2009). Furthermore, we provide a biclustering method that simply uses localization followed by a random extraction of submatrices from the provided initial configuration. We show that this method when applied on real data outperforms the tested greedy heuristics according to certain evaluation criteria.

2 METHODS AND ALGORITHMS

Given an input data matrix M where rows correspond to genes, columns correspond to conditions, and each data entry is a real value corresponding to the expression change of the gene under the specified condition, the idea behind *localization* is to first group together rows and columns of M in such a way that correlated rows/columns are 'localized', that is rows/columns exhibiting similar patterns are placed in nearby locations within M. In order to formalize this notion, first a decision has to be made with regards to where in matrix M to collect entries with similar patterns. Gathering such patterns around the main diagonal of M seems like a natural choice. Let $a_{i,j}$ denote the real valued data entry at row i and column j of M. Formally, the goal of localization is to reorder the rows and columns of M in such a way as to minimize $\sum_{\forall i,j} a_{i,j} \times |i-j|$. We note that such an optimization in 0–1 matrices corresponds to row/column permutations that bring non-zero entries as close to the diagonal as possible. This optimization problem has received considerable attention from a wide range of application areas including graph drawing, VLSI layout and numerical analysis under different names such as bipartite linear arrangement, minimum-1-sum, bandwidth sum, edge sum, wirelength minimization; see Díaz *et al.* (2002); Lai and Williams (1999) for nice surveys on the topic.

Algorithm 1 Localization

repeat
  `/* Fix column set C, order row set R */`
  for all $r \in R$ **do**
    $leftsum = 0$; $rightsum = \sum_{i=2}^{|C|} a_{r,i}$;
    for $c = 1$ **to** $|C|$ **do**
      if $leftsum \geq rightsum$ **then** break;
      $leftsum += a_{r,c}$; $rightsum -= a_{r,c+1}$;
    end for
    $P_c = P_c \cup \{r\}$;
  end for
  for all $c \in C$ **do**
    Let $u, v \in R$. We define a total order $u \preceq v$ as follows:
    $\sum_{i=1}^{c} a_{v,i} \times \sum_{i=c}^{|C|} a_{u,i} \leq \sum_{i=1}^{c} a_{u,i} \times \sum_{i=c}^{|C|} a_{v,i}$
    Sort P_c using ordering defined by $\preceq$.
  end for
  `/* Fix row set R, order column set C */`
  Switch R and C, and repeat the above procedure.
until no change in row/column ordering or enough iterations

2.1 The localization algorithm

Since the optimization goal determined for localization is computationally hard in general, it is of interest to establish connections with other well-studied layout problems. In Shahrokhi *et al.* (2001), it has been theoretically shown that the bipartite linear arrangement and the problem of minimizing crossings in a bipartite drawing is closely related and that the relation is approximation preserving: given an $f(.)$ approximation for one of the problems, where f is some function on the input size, it also provides an approximation for the other with the same ratio. This connection has also been verified in practice through various experimental evaluations (Stallmann *et al.*, 2001). Based on these results we present our localization method that aims at minimizing:

$$\sum_{\forall i,j} a_{i,j} \times (L_{i,j} + R_{i,j}) \tag{1}$$

where $L_{i,j} = \sum_{(p>i,q<j)} a_{p,q}$ and $R_{i,j} = \sum_{(p<i,q>j)} a_{p,q}$. Note that considering the partitions of a bipartite graph as rows and columns, and the weights of the edges as the real valued entries in the data matrix, minimizing weighted edge crossings in the bipartite graph is equivalent to minimizing Equation (1) in a data matrix. A pseudocode of the localization method is provided in Algorithm 1. Assuming a fixed arrangement of columns, first the rows are reordered so as to reduce the sum in Equation (1). Then the rows are kept fixed and the columns are reordered using the same procedure. This is continued until no further improvements can be made or a predetermined number of iterations is reached. For rearranging one of the dimensions while

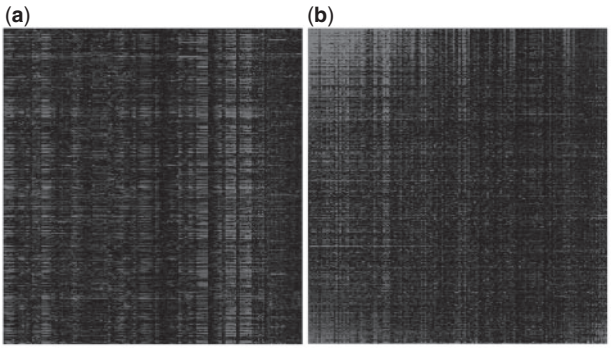

Fig. 1. Yeast expression data (Gasch *et al.*, 2000) (**a**) Before Localization. (**b**) After Localization.

fixing the other, we adopt our method presented in Çakiroglu *et al.* (2009) which has been suggested in a different context with a similar optimization goal. For completeness, we provide the proof of the following lemma in the Supplementary Material.

LEMMA 2.1. *Assuming the columns (rows) are fixed, the localization algorithm orders the row (column) set in such a way that the sum in Equation (1) is at most three times the optimum among all possible row (column) orderings of the matrix.*

With a straightforward implementation, the running time of each iteration of the *repeat loop* is $O(|R|\times|C|+|R|\log|R|+|C|\log|C|)$, where $|R|,|C|$ denote the sizes of the rows and the columns of the data matrix respectively. The loop is iterated until the minimization of Equation (1) converges or a constant number of iterations is achieved. We note that in all the experiments conducted, the method converges after a small constant number of iterations. In most practical settings $|C|=O(|R|)$, that is the number of conditions is usually much smaller than the number of genes and $|C|=\Omega(\log|R|)$. Therefore, under these settings, the running time of the proposed method is linear in terms of the input size. Heatmap visualizations of the *Yeast* expression data both before and after localization are provided in Figure 1. The Figure 1a shows the original gene expression data matrix where the data values are assigned colors for visualization purposes. The Figure 1b shows the gene expression matrix after the localization method is applied. The right block gathering homogeneous colored subblocks together as compared to the left block is a visual clue that the localization method does 'localize' the matrix by grouping together the submatrices with correlated gene expression data values.

2.2 Improving biclustering heuristics via localization

The most natural heuristic candidates to apply localization as a pre-processing step for further improvement are those based on iteratively improvin some locally optimum solution in a greedy fashion. We identify three such algorithms, briefly describe each, and comment on possible modifications necessary for pre-processing with our localization method.

The SAMBA (Tanay *et al.*, 2002) algorithm identifies biclusters using an exhaustive enumeration method. First, a bipartite graph model of the gene expresssion matrix is created. The set of genes and the set of conditions correspond to the two partitions and there exists an edge between a gene–condition pair if the expression level of the gene changes significantly under the given condition. Weights are assigned to the edges and non-edges in the bipartite graph in a way that the weight of a subgraph corresponds to its statistical significance. Once the weights are assigned the problem then turns into that of extracting maximum weight complete subgraphs. The subgraph construction phase is followed by an iterative local improvement procedure of growing/shrinking the composed subgraphs. Localization pre-processing becomes affective at this phase by localizing the vertices corresponding to coexpressed genes and thus providing a better chance for local improvement at each iteration.

MSBs (Liu and Wang, 2007) are yet another algorithm we envision improvement via localization pre-processing. In the original description of the algorithm, first a similarity matrix for a given reference gene is constructed. Each entry in the similarity matrix descibes how similar the expression value of a gene under a certain condition is to the expression value of the reference gene under the same condition. It is shown that a greedy iterative strategy of removing the least similar row or column from the similarity matrix yields the MSB. To generalize this approach for the extraction of all the biclusters in the expression matrix a subset of reference genes are needed. This is achieved via randomly selecting a subset of genes and designating them as reference. The new algorithm Randomized Maximum Similarity Biclusters (RMSBs) is the resulting algorithm. As each output bicluster is computed based on the given reference gene it is important to carefully construct the set of references. We extend this biclustering algorithm by pre-processing the data matrix with localization and selecting a reference gene subset by considering genes at constant intervals $gene_i, gene_{i+\epsilon}, gene_{i+2\times\epsilon}, \ldots gene_{i+k\times\epsilon}$. It should be intuitive that selecting a gene from each local neighborhood in the localized matrix would yield better references than selecting them in random.

LAS proposed by Shabalin *et al.* (2009) is yet another recent algorithm for which we utilize the localization pre-processing for further improvement. Assuming a Gaussian null model for the data, the significance score of a submatrix is defined using a Bonferroni-corrected P-value based on the size and the average of the data values in the submatrix. The main goal then is defined as that of extracting the submatrix with maximum score. As with the previous algorithms this optimization goal is computationally hard to achieve. Therefore, rather than solving it to the optimum a greedy iterative heuristic is proposed for the search procedure. The heuristic starts out with a random initial submatrix. Fixing alternatively the column (row) set, the set of rows (columns) locally optimizing the significance score is searched iteratively until convergence. Pre-processing the LAS algorithm with localization and modifying the greedy search heuristic to start the iterations with a submatrix consisting of rows and columns within a local neighborhood rather than a random submatrix yields an improvement in the local search results.

2.3 Random extraction after localization

In order to demonstrate the power of the localization procedure, we provide a simple algorithm that takes as input the localized matrix M_L, extracts local submatrices randomly from M_L, evaluates the significance (statistical and/or biological) score of each submatrix, and finally reports those with high scores.

Given minimum and maximum row counts, and a row increment amount respectively denoted with min_r, max_r, inc_r the set of possible row sizes are $min_r + k\times inc_r$, for all non-negative integers k where $k\times inc_r \le max_r$. The set of possible column sizes is defined analogously. The dimension of each extracted local submatrix is a member of the cartesian product of these two sets. For each possible dimension we extract a pre-specified number of submatrices randomly and eliminate those with low significance. The rows (columns) of the submatrix are all consecutive in M_L. We evaluate the significance of each extracted submatrix with two types of scorings. First is the H-value test (Cheng and Church, 2000). Let the set of rows and columns of the extracted submatrix be denoted respectively with I, J. The residue R of the entry (i,j) is

$$R_{I,J}(i,j) = a_{i,j} - a_{Ij} - a_{iJ} + a_{IJ} \tag{2}$$

where a_{iJ} is the mean of row i, a_{Ij} is the mean of column j and a_{IJ} is the mean of the submatrix. H-value of the submatrix is then defined as,

$$H-value(I,J) = \frac{1}{|I||J|}\sum_{i=0,j=0}(R_{I,J}(i,j)^2) \tag{3}$$

The submatrices with H-value lower than a given threshold η are subject to a second test when biological validation data in the form of GO associations

through sources such as FuncAssociate (Berriz *et al.*, 2003) is avaliable. Given such associations it is easy to determine the ratio of the number of genes associated with some category to the total number of genes in the extracted submatrix. If the highest ratio of the submatrix is lower than a threshold the submatrix is eliminated, otherwise it is reported as a bicluster with high significance. In what follows random extraction after localization is referred to as *REAL* if only H-value scoring is applied. When biological significance is also evaluated with available GO associations we refer to the algorithm as $REAL_{GO}$.

3 DISCUSSION OF RESULTS

The localization algorithm and the bicluster extraction algorithm REAL are implemented in C++ using the LEDA Library (Mehlhorn and Naher, 1999). The codes for the algorithm implementation and experimentation are available as part of the Supplementary Materials. The implementations of the algorithms subject to improvement via localization, SAMBA, MSB and LAS are from the cited sources of the algorithms. Each of these algorithms are evaluated both with and without localization pre-processing. In the former case, the algorithms are denoted with SAMBA*, MSB* and LAS*. We experiment on two real datasets, *Saccharomyces cerevisiae* (*Yeast*) from Gasch *et al.* (2000) and *A.thaliana* from Wille *et al.* (2004). We note that our localization algorithm requires non-negative real values in the input matrix. To achieve this, we apply a normalization procedure on the input expression matrix by first adding the minimum value to each entry and then taking the logarithm of the data entries in the resulting matrix. The logarithm transformation is a standard transformaton in microarray data analysis to achieve distributions that are closer to normal distributions (Kluger *et al.*, 2003; Liu and Wang, 2007). We emphasize that to have a fair comparison, right after localization we preserve the original values of the expression matrix before we apply the algorithms SAMBA, MSB and LAS, though localization may have changed the order of the rows and the columns.

3.1 Structural and statistical evaluation

We first evaluate the structural and statistical significance of biclusters output by each algorithm REAL, $REAL_{GO}$, MSB, SAMBA, LAS and the localization pre-processed counterparts of the last three. The parameter settings of each of these algorithms and those of their localization pre-processed counterparts are the same as the default settings suggested in the original descriptions of the methods. The results of our evaluations applied on both *A.thaliana* and the Yeast gene expression datasets are presented in Table 1. We apply REAL on the Thaliana dataset and $REAL_{GO}$ on the Yeast dataset. The parameter η is set to 75 for REAL and to 0.12 for $REAL_{GO}$. The parameter settings for each algorithm are the same in this and the following experimental subsections. In addition to structural information including maximum and minimum bicluster sizes, average dimensions and the bicluster count we also provide evaluations based on statistical measures of similarity. Candidate measures commonly used in literature are H-value (Cano *et al.*, 2007; Cheng and Church, 2000; Cheng *et al.*, 2008; Liu *et al.*, 2009), *Hv-value* (Bryan and Cunningham, 2006), average correlation value (ACV) (Teng and Chan, 2006). All these measures have similar definitions and provide similar results in our evaluations, and we only provide the results of H-value evaluations.

Table 1. Structural and statistical analysis of biclusters produced by the algorithms under consideration applied on the *Arabidopsis thaliana* (top multirow) and the yeast (bottom multirow) datasets

Alg.	Max	Min	Avg $\|I\|$	Avg $\|J\|$	Count	H-value
REAL	180×5	20×5	43.85	7.52	143	54.6
MSB	319×163	5×6	58.00	22.00	7	411.7
MSB*	69×55	4×5	23.69	21.58	36	0
SAMBA	60×4	12×4	18.85	4.22	55	2810.7
SAMBA*	32×4	6×6	17.48	4.63	52	2062.5
LAS	7×3	1×1	6.10	1.22	60	7693.7
LAS*	14×1	1×1	5.73	1.28	60	15007.6
$REAL_{GO}$	47×30	5×10	24.10	14.50	534	0.107
MSB	112×111	11×11	20.35	19.35	1342	0.241
MSB*	105×106	11×11	19.11	18.29	1424	0.306
SAMBA	174×71	7×10	64.94	16.71	114	0.474
SAMBA*	119×71	19×6	63.08	16.19	128	0.524
LAS	679×113	1×1	134.82	23.91	55	0.388
LAS*	688×111	1×1	120.24	22.17	56	0.386

At the *A.thaliana 734x69* major row, the best performers in terms of H-values are the algorithms REAL and MSB*. MSB* interestingly provides several constant-valued biclusters as its H-value average is 0. Localization pre-processing helps MSB provide more constant biclusters than the original algorithm. Comparing SAMBA and SAMBA*, the localization pre-processed SAMBA* provides more homogeneous biclusters in terms of H-value. LAS and LAS* are among the worst ones. This is because of the high variance of the dataset. Also, due to the scoring scheme of the LAS algorithm both the original algorithm and the localization pre-processed modification provide relatively small biclusters. When we compare the original and localized runs of the algorithms on the *Yeast 2993x173* dataset, we note a slight increase in the H-values of MSB* and SAMBA* as compared to their unlocalized counterparts. However the localization pre-processed versions of the algorithms provide a larger number of biclusters in each case which maybe the cause of this discrepancy.

3.2 Comparison with localization alternatives

We compare the performance of our localization algorithm with another potential pre-processing method, the non-smooth non-Negative Matrix Factorization (nsNMF) proposed by Carmona-Saez *et al.* (2006). This method is not originally proposed as a pre-processing method *per se*, but rather as a full-fledged biclustering method. However since the main idea involves reordering the rows and the columns of the data matrix so as to gather correlated structures in close proximity, it serves as a potential benchmark[1] method for pre-processing.

In nsNMF, the input data matrix M is first converted to a product of two matrices W and H with sizes $|R| \times f$ and $f \times |C|$, respectively. Each of the f columns of W is called a *factor* and

[1]Another feasible alternative would be running the algorithms under consideration on several randomly permuted instances of the input and picking the 'best' output instance rather than employing a pre-processing of some sort. We show this alternative does not improve the biclustering results by running each algorithm on 50 randomly ordered instances of the *S.cerevisiae* dataset. Details can be found in the Supplementary Material.

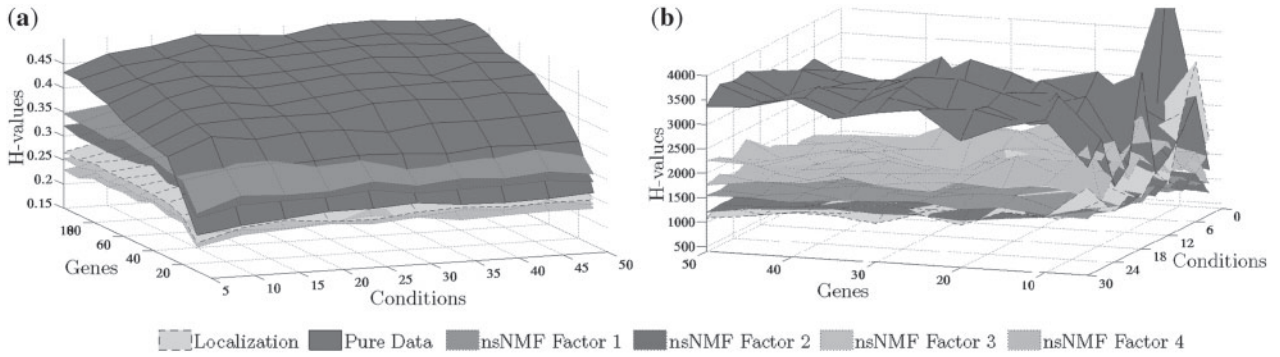

Fig. 2. (**a**)*Yeast* data 2993 genes and 173 conditions and plots corresponds to H-values; (**b**)*A.thaliana* data 734 genes and 69 conditions and plots corresponds to H-values.

constitutes a basis experiment. Each row of H constitutes a basis gene. Given a certain factor, that is a column of W, the rows of M can be sorted based on it. Similarly, the columns of M are sorted according to the corresponding row, that is the basis gene of H. It is assumed that each such sorting of the original data matrix provides correlated genes/conditions localized in a portion of the matrix. For our evaluations we assume the parameter settings employed in the original paper. For nsNMF we produce results for four factors. Therefore we evaluate six matrices in total: M, M_L, M_1, M_2, M_3, and M_4. The last four correspond to the results of nsNMF with basis factorizations shown as the corresponding subscripts.

We present 3D plots in Figure 2a and b which summarize the results of our experiments performed on the Yeast expression data (Gasch *et al.*, 2000) and the *A.thaliana* expression data (Wille *et al.*, 2004) respectively. We use the H-value test described in Equation 3 to compare the localization performances of the six alternative matrices. Each point on the x-axis corresponds to a specific gene size g_s where $g_s \in \{10, 20, 30, \ldots, 100\}$ for the Yeast expression data experiment and $g_s \in \{5, 10, 15, \ldots, 50\}$ for the *A.thaliana* expression data experiment. Each point on the y-axis corresponds to a specific condition size c_s where $c_s \in \{5, 10, 15, \ldots, 50\}$ for the Yeast data and $c_s \in \{3, 6, 9, \ldots, 30\}$ for the Thaliana data. For each pair (g_s, c_s), we randomly pick a submatrix with g_s consecutive rows and c_s consecutive columns from each of the six matrices starting at exactly the same indices and compute an H-value of the submatrix. The random choices are repeated 1000 times, an average H-value is computed for each alternative and is plotted at the z-axis. Analyzing the results of our experiments on the Yeast dataset, H-value test seems to favor M_L, the matrix pre-processed with our Localization procedure over M, M_1, M_2, the original input matrix, and the resulting matrices of nsNMF with basis factorizations 1 and 2, respectively. We note that low H-value implies more correlation between the entries of a given submatrix. The factorizations with basis 3 and 4 seem to provide H-values slightly better than that of localization. With regards to the *A.thaliana* dataset experiments H-value results of localization is better than the rest in almost all the cases, except for the case where g_s and c_s are small. In this exceptional case, the H-value test is not stable due to the large variance in the original dataset. As alternative statistical significance measures the six matrices are also evaluated based on variances and the average pearson correlation coefficient (PCC) ratios. Such an evaluation reveals the intuition behind the nsNMF method and its contrast with the localization. Detailed 3D plots and a discussion of these results emphasizing the main conceptual differences between nsNMF and localization can be found in the Supplementary Material.

Although they do not provide an explicit running time, the algorithm of Carmona-Saez *et al.* (2006) seems to require $\Omega(|R|^2 \times |C| \times f \times I_{conv})$ time, where I_{conv} is the number of iterations necessary for convergence to local optimum. In practice, the constant in the suggested bound is large and the running time is much worse. The fact that the greedy iterative biclustering methods themselves require large amounts of computing time, necessitates a pre-processing algorithm for improvement of such methods take much less proportion of the necessary CPU time. This makes nsNMF a much less appealing pre-processing alternative. Our localization algorithm provides better or comparably similar results to those of nsNMF with only an almost linear running time.

3.3 Functional enrichment evaluation

We next evaluate the biclustering results of the algorithms under study based on biological significance. Based on gene functionality a gene-to-category association is created for the Yeast dataset in von Mering *et al.* (2002). Using these associations, we are able to evaluate an *enrichment ratio* for each bicluster. Categories of genes are identified in a manner similar to Bryan and Cunningham (2006). There are 13 pre-identified categories. Given a bicluster the ratio of the number of genes specified in a category to the number of genes in the bicluster provides a possible enrichment value for that category. For a specific category, we choose the highest enrichment value among all biclusters as the *enrichment ratio* of the category. This is a ratio between 0 and 1, see Table 2. For this experiment, we do not evaluate small biclusters that contain less than 20 genes.

We first compare the results of each greedy iterative heuristic method with those of its localization pre-processed counterpart. MSB* provides better enrichment ratio than MSB in 10 categories. MSB has better ratio than MSB* in only one category and they have a tie in two of the categories. These are the expected results as the MSB algorithm greedily selects the maximum similarity bicluster based on a random reference gene, whereas with the localization pre-processing each reference gene is selected from a representative local neighborhood that already satisfies a certain degree of similarity. The contrast is also apparent in the comparison

Table 2. Warfield for the *Yeast 2993x173* dataset

	Original Algorithms				Localized Runs		
Func.	REAL$_{GO}$	MSB	SAMBA	LAS	MSB*	SAMBA*	LAS*
E	0.58	0.15	0.40	0.56	**0.25**	**0.53**	**0.63**
G	*0.73*	**0.19**	0.39	0.42	0.18	0.39	**0.49**
M	*0.84*	0.31	0.33	**0.39**	**0.38**	**0.35**	0.29
P	0.81	0.29	0.89	**0.89**	**0.36**	**0.92**	0.77
T	*0.74*	0.33	0.41	**0.37**	**0.41**	**0.46**	0.33
B	*0.38*	0.29	0.21	0.15	**0.33**	**0.25**	**0.16**
F	0.35	0.27	**0.71**	**0.24**	**0.32**	0.49	0.19
O	*0.74*	0.26	**0.26**	0.17	0.26	0.20	0.17
A	*1.00*	0.25	**0.47**	0.43	**0.29**	0.38	**0.44**
R	0.09	0.13	0.20	0.24	**0.17**	**0.27**	**0.30**
D	*0.71*	0.33	**0.24**	0.25	**0.35**	0.23	**0.27**
C	0.26	0.33	**0.44**	0.29	0.33	0.41	**0.30**
U	*0.37*	0.24	0.16	**0.17**	**0.33**	**0.28**	0.13

Predefined functional categories; E, energy production; G, Amino Acid Metab., M, Other Metab., P, Translation; T, Transcription; B, Transcriptional control; F, Protein Fate; O, Cellular Org.; A, Transport and Sensing; R, Stress and Defense; D, Genome Maintenance; C, Cellular Fate / Org.; U, Uncharacterized. For each pair of original algorithm and localization pre-processed counterpart a bold value highlights larger enrichment ratio.

of SAMBA with its localized counterpart SAMBA*. The localization pre-processed version provides a better functional enrichment ratio than the original algorithm in seven categories, whereas the original algorithm beats the modified version in five categories. There is a draw in one category. A similar contrast results when comparing LAS versus LAS*. As a result as far as the functional enrichment is concerned, the localization pre-processed counterparts of all three greedy iterative biclustering methods provide more significant biclusters than the original methods themselves.

We note that with this measure the best performer among all the algorithms, localization pre-processed or not, is the simple random extraction algorithm applied after localization, that is REAL$_{GO}$. It provides better enrichment ratio than the rest of the six algorithms in eight categories. The category of *Transport and Sensing* is especially notable as REAL$_{GO}$ provides a bicluster with all the genes associated with that category giving rise to a functional enrichment ratio of exactly one for that category. This is a major indication of success especially given that the average number of genes in a bicluster produced by REAL$_{GO}$ is almost 24, see Table 1. Both the localization pre-processing and the GO association validation of the randomly extracted structures contribute to this achievement. We note that the results presented in Table 2 are verified when the enrichment ratios are calculated based on Bonferroni-corrected *P*-values as well. Details of this evaluation can be found in the Supplementary Material.

3.4 Protein protein interactions evaluation

There is usually a natural interconnection between the information contained within expression data and protein-protein interaction (PPI) networks (Jansen *et al.*, 2002). The validity of high-level structures such as biclusters is usually cross-checked via low-level biological data in the form of protein interactions (Prelić *et al.*, 2006). Conversely, to improve the accuracy and the coverage of predicted protein interactions it is usually necessary to integrate complementary data sources including gene expression (Lin *et al.*, 2009). Therefore, our next evaluation is based on validating extracted biclusters with known protein–protein interactions from literature.

Table 3. PPI experiment on *Yeast* and *A.thaliana* datasets with PPIs

	REAL$_{GO}$	MSB	MSB*	SAMBA	SAMBA*	LAS	LAS*
σ_1	0.20286	0.2049	**0.2255**	0.1088	**0.1234**	**0.1147**	0.0996
σ_2	0.15341	0.1494	**0.1697**	0.0716	**0.0821**	0.0551	**0.0640**
σ_3	0.00442	0.0016	**0.0037**	0.0167	**0.0199**	0.0000	0.0000

Averages of all hit ratios are given as σ_1 for PPI$_1$ (23 702 interactions), σ_2 for PPI$_2$ (11 833 interactions) and σ_3 for PPI$_3$ (24 418 interactions). For each pair of original algorithm and localization pre-processed counterpart a bold value highlights larger hit ratio.

We make use of two separate sources of PPI network data for the yeast, one from Ben-Hur and Noble (2005) with 23 702 interacting gene pairs and the other from Suthram *et al.* (2006) with 11 833 interacting gene pairs. As for the *A.thaliana* dataset evaluations we use the PPI network with 24 418 interactions and 11 706 genes from De Bodt *et al.* (2009). For the rest of the evaluations, these data sources are denoted with PPI$_1$, PPI$_2$, PPI$_3$, respectively.

For each bicluster extracted using a specified algorithm, we define the *hit ratio* of the bicluster as the ratio of the number of interactions among the gene pairs in the bicluster to the square of the total number of genes in the bicluster. Extracting the subnetwork of genes of the bicluster from an appropriate PPI network, the hit ratio in other words is a measure of the density of this subnetwork. Therefore, given an algorithm and a PPI network source it is easy to compute the hit ratios of all the biclusters produced by the algorithm. We compute an average σ of the hit ratios per algorithm for our evaluations, see Table 3. As far as the protein–protein interactions of the given source are concerned, the higher the average hit ratio, the more correlated the gene groups in output biclusters are.

With regards to PPI$_1$ network data localization pre-processing improves the results of the MSB and the SAMBA algorithms, but fails to improve those of LAS. An increase in the average hit ratio of the produced biclusters is assumed to be an improvement in this case. Localization pre-processing is verified even more strongly in the PPI$_2$ network data, as all the modified versions of the algorithms provide better hit ratios than their original counterparts. We note that going from PPI$_1$ to PPI$_2$ the average hit ratios decrease as the former is more dense than the latter. As for the PPI$_3$ network data as the network is the sparsest among all three networks the average hit ratios are all quite low. However, it is still easy to verify the improvement created by localization pre-processing. Overall, three network evaluations MSB* and the REAL$_{GO}$ algorithms are among the best performers as far as the average hit ratios of the produced biclusters are of concern.

ACKNOWLEDGEMENTS

We would like to thank A. Aladag for his help with the implementations.

Funding: The Scientific and Technological Research Council of Turkey (TUBITAK; 109E071) in part; The Scientific and

Technological Research Council of Turkey (TUBITAK, BIDEB; 2211 to M.S.).

Conflict of Interest: none declared.

REFERENCES

Abdullah,A. and Hussain,A. (2006) A new biclustering technique based on crossing minimization. *Neurocomputing*, **69**, 1882–1896.

Alexe,G. *et al*. (2004) Consensus algorithms for the generation of all maximal bicliques. *Disc. Appl. Math.*, **145**, 11–21.

Barkow,S. *et al*. (2006) Bicat: a biclustering analysis toolbox. *Bioinformatics*, **22**, 1282–1283.

Ben-Dor,A. *et al*. (2002) Discovering local structure in gene expression data: the order-preserving submatrix problem. In *Proceedings of the International Conference on Computing Biology, RECOMB '02*, ACM, New York, NY, USA, pp. 49–57.

Ben-Hur,A. and Noble,W.S. (2005) Kernel methods for predicting protein-protein interactions. *Bioinformatics*, **21**(Suppl. 1), i38–i46.

Bergmann,S. *et al*. (2003) Iterative signature algorithm for the analysis of large-scale gene expression data. *Phys. Rev. E, Stat., Nonlinear, Soft Matter Phys.*, **67** (3 Pt 1), 031902.

Berriz,G.F. *et al*. (2003) Characterizing gene sets with funcassociate. *Bioinformatics*, **19**, 2502–2504.

Bryan,K. and Cunningham,P. (2006) Bottom-up biclustering of expression data. In *IEEE Symposium on Computing Intelligence and Bioinformatics and Computing Biology*, Toronto, Ontario, pp. 1–8.

Cano,C. *et al*. (2007) Possibilistic approach for biclustering microarray data. *Comp. Biol. Med.*, **37**, 1426–1436.

Carmona-Saez,P. *et al*. (2006) Biclustering of gene expression data by non-smooth non-negative matrix factorization. *BMC Bioinformatics*, **7**, 78.

Çakiroglu,O.A. *et al*. (2009) Crossing minimization in weighted bipartite graphs. *J. Disc. Algs*, **7**, 439–452.

Cheng,K.-O. *et al*. (2008) Identification of coherent patterns in gene expression data using an efficient biclustering algorithm and parallel coordinate visualization. *BMC Bioinformatics*, **9**, 210.

Cheng,Y. and Church,G.M. (2000) Biclustering of expression data. In Bourne,P.E. *et al*. (eds) *Proceedings of the 8th International Conference on Intelligent Systems for Molecular, ISMB'00)*, AAAI Press, Menlo Park, CA, pp. 93–103.

De Bodt,S. *et al*. (2009) Predicting protein-protein interactions in arabidopsis thaliana through integration of orthology, gene ontology and co-expression. *BMC Genomics*, **10**, 288.

Díaz,J. *et al*. (2002) A survey of graph layout problems. *ACM Comput. Surv.*, **34**, 313–356.

Erten,C. and Sözdinler,M. (2009) Biclustering expression data based on expanding localized substructures. In *Proceedings of the 1st International Conference on Bioinformatics and Computing Biology*, Springer, Berlin, Heidelberg, pp. 224–235.

Gasch,A.P. *et al*. (2000) Genomic expression programs in the response of yeast cells to environmental changes. *Mol. Biol. Cell*, **11**, 4241–4257.

Getz,G. *et al*. (2000) Coupled two-way clustering analysis of gene microarray data. *Proc. Natl Acad. Sci. USA*, **97**, 12079–12084.

Hartigan,J.A. (1972) Direct clustering of a data matrix. *J. Am. Stat. Assoc.*, **67**, 123–129.

Jansen,R. *et al*. (2002) Relating whole-genome expression data with protein-protein interactions. *Genome Res.*, **12**, 37–46.

Kluger,Y. *et al*. (2003) Spectral biclustering of microarray data: coclustering genes and conditions. *Genome Res.*, **13**, 703–716.

Lai,Y.L. and Williams,K. (1999) A survey of solved problems and applications on bandwidth, edge-sum, and profile of graphs. *J. Graph Theory*, **31**, 75–94.

Lin,X. *et al*. (2009) Assessing reliability of protein-protein interactions by integrative analysis of data in model organisms. *BMC Bioinformatics*, **10** (Suppl. 4), S5.

Liu,J. *et al*. (2009) Biclustering of microarray data with mospo based on crowding distance. *BMC Bioinformatics*, **10** (Suppl. 4), S9.

Liu,X. and Wang,L. (2007) Computing the maximum similarity bi-clusters of gene expression data. *Bioinformatics*, **23**, 50–56.

Madeira,S.C. and Oliveira,A.L. (2004) Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, **1**, 24–45.

Mehlhorn,K. and Naher,S. (1999) *Leda: A Platform for Combinatorial and Geometric Computing*. Cambridge University Press, Cambridge.

Murali,T.M. and Kasif,S. (2003) Extracting conserved gene expression motifs from gene expression data. In *Proceedings of the 8th Pacific Symposium on Biocomputing Lihue, Hawaii*, pp. 77–88.

Prelić,A. *et al*. (2006) A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics*, **22**, 1122–1129.

Shabalin,A.A. *et al*. (2009) Finding large average submatrices in high dimensional data. *Ann. Appl. Stat.*, **3**, 985.

Shahrokhi,F. *et al*. (2001) On bipartite drawings and the linear arrangement problem. *SIAM J. Comput.*, **30**, 1773–1789.

Sharan,R. *et al*. (2003) Click and expander: a system for clustering and visualizing gene expression data. *Bioinformatics*, **19**, 1787–1799.

Srinivas,M. and Patnaik,L. (1994) Genetic algorithms: a survey. *Computer*, **27**, 17–26.

Stallmann,M. *et al*. (2001) Heuristics, experimental subjects, and treatment evaluation in bigraph crossing minimization. *J. Exp. Algorithmics*, **6**, 8.

Suthram,S. *et al*. (2006) A direct comparison of protein interaction confidence assignment schemes. *BMC Bioinformatics*, **7**, 360.

Tanay,A. *et al*. (2002) Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, **18** (Suppl. 1), S136–S144.

Teng,L. and Chan,L.-W. (2006) Biclustering gene expression profiles by alternately sorting with weighted correlated coefficient. In *Machine Learning for Signal Processing, 2006. Proceedings of the 2006 16th IEEE Signal Processing Society Workshop on*, Arlington, VA, pp. 289–294.

von Mering,C. *et al*. (2002) Comparative assessment of large-scale data sets of protein-protein interactions. *Nature*, **417**, 399–403.

Wille,A. *et al*. (2004) Sparse graphical gaussian modeling of the isoprenoid gene network in arabidopsis thaliana. *Genome Biol.*, **5**, R92.