Silesian University of Technology
Faculty of Automatic Control, Electronics and Computer Science
Institute of Informatics

Doctor of Philosophy Dissertation

Bi-clustering – algorithms and applications

Paweł Foszner

Supervisor: prof. dr hab. inż. Andrzej Polański

Gliwice, 2014

To my lovely wife Aleksandra for her full support over those years.

Table of Contents

Acknowledgements

I would like to thank my supervisor for his understanding and patience. Without his help, this thesis would never have been written. I would also like to thank Aleksandra Gruca. She helped me to start the Ph.D. and helped step by step with the very first publication. Her guidance and advice were very useful during the whole studies.

This work was supported by the European Union from the European Social Fund (grant agreement number: UDA-POKL.04.01.01-00-106/09).

1. Introduction

Nowadays, we observe still very rapid development in the field of telemetry, biomedical analysis, text mining, data mining in general and many others. As a result of these studies we usually get very large and complex data sets. Classical approaches such as clustering, can extract only part of the relevant information. For example for gene expression data, which contain information about the expression of genes under different conditions, using simple clustering approach we are able to find groups of genes that reveals similar expression under all conditions. Figure 1 shows comparison between clustering (A) and bi-clustering (B). Even those techniques find the same cluster of genes, clustering technique lose information about conditions under which this group is co-expressed.

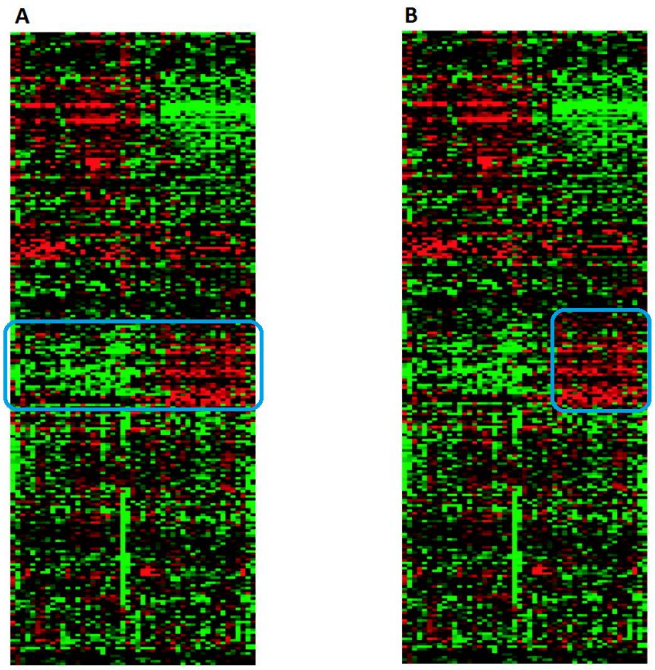


Figure 1. Comparison between classical clustering approach versus bi-clustering.

Bi-clustering is a technique that in two-dimensional data finds a subset of attributes from one dimension that reveals similar behavior only on subset of attributes from second dimension.

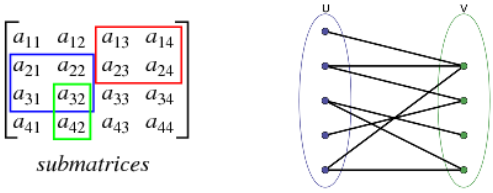


Figure 2. Simple visualization of bi-clustering.

In a very simple words bi-clustering is about finding sub-matrices in data matrix or finding a bi-cliques in bipartite graphs (as it is shown on Figure 2).

Bi-clustering is a data mining technique which allows simultaneous clustering of the rows and columns of a matrix. This technique belongs to the class of NP-hard problems, and was first presented by Morgan and Sonquist in 1963 [1], than by Hartigan [2](1972), and by Mirkin in 1996 [3]. In the context of bioinformatics problems, the first to use this technique was Cheng and Church [4]. They proposed bi-clustering of result from microarray experiments, by introducing the mean square residue in bi-cluster. Representative of modern algorithms can be QUBIC, introduced by the Guojun Li, et al [5]. They proposed very efficient algorithm for bi-clustering of matrix with discretized expression data. Authors use graph representation of data, and like Cheng and Church, also find bi-clusters with low mean square residue. Over the years, since the publication of Morgan to this day, has raised a number of different approaches to bi-clustering. The methods differ from each other both in the approach to modeling the input data (bipartite graph [5], discrete matrix [6] trees [7]), and also the way of obtaining the final results (exhaustive search [5], the decomposition of the matrix [8], the search graph [2]).

2. Aims

As it is shown in more detail in chapter 3 we can distinguish multiple classification of bi-clusters regarding to its structure or position in data matrix. Each case may need a different approach. The task of selecting the appropriate method requires a very good understanding of the data to be analyzed. A very difficult task, as complex as the task of choosing the appropriate number of bi-clusters (the number of which is often the input parameter for many bi-clustering algorithms). The algorithm of processing data in bi-clustering may look as presented on Figure 3.

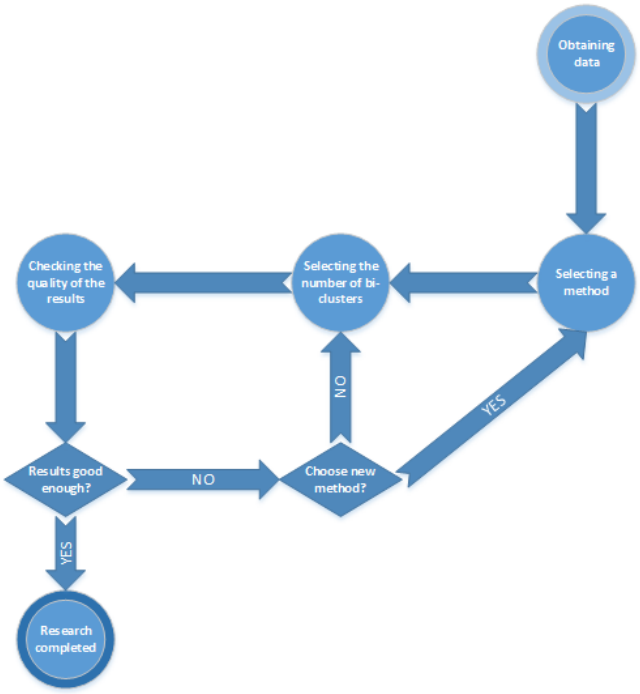


Figure 3. Bi-clustering analysis sample workflow.

We are never able to say with absolute certainty that we have data containing bi-clusters of a certain structure. Therefore, the process of obtaining bi-clusters is always an iterative process. Each iteration includes activities related to the selection of parameters, and very often an attempt to determine the number of bi-clusters. Each of these steps is usually performed manually by the scientist. He will be responsible for excellent knowledge of the analyzed data.

The aims of the thesis were:

- To implement all major literature algorithms for data bi-clustering.
- To apply implemented algorithms to both artificially created and real datasets.
- To develop methodologies for comparisons of different bi-clustering algorithms and to draw conclusions stemming from using these methodologies for artificial and real datasets.
- To introduce improvements in bi-clustering ideas. The main improvement proposed in the thesis was an algorithm with can be applied to any type of data with any bi-cluster structure. The proposed algorithm is a meta-algorithm, which uses the ensemble methodology ideas. Later in the thesis, it is proven that ensemble approaches relying on the combination of results of different algorithms (specialized for various applications) will make quality of outcome resistant to bi-cluster structure.

The strategy of the performed research was oriented towards simplifying the analysis of bi-clustering to a pipeline as simple as possible: providing data on the input and getting the results on the output (Figure 4). The role of the user in this system is limited to the loading on the input data. However, it may also adjust the parameters used in the analysis.

Figure 4. Simplified bi-clustering analysis workflow.

The key idea of proposed method is to computed large number of bi-clustering algorithms, each of which is specialized in different kinds of bi-clusters. Algorithms used in my analysis are described in Chapter 6. Then, the results of these methods are combined into one. For this purpose have been proposed similarity measure (Chapter 7.1) of single bi-clusters and modification of the Hungarian algorithm for pairing them (Chapter 7.3).

When paired, as the results we obtain set of sets of bi-clusters. Within a single set of bi-clusters, last step of algorithm is to connect bi-clusters composing it to a single one (Chapter 7.4).

3. Theses

On the basis of the research performed the following these are claimed:

1. The elaborated methodology for comparisons of results of bi-clustering, based on generalized Munkres algorithm, is an efficient and flexible tool well suited for analyzes of real datasets.
2. The elaborated meta – bi-clustering algorithm improves performance of bi-clustering.

4. Main contribution and original elements of the thesis

This work is development of the work carried out during last five years. First large scale analysis were performed in 2009 on data from tumor tissue bank of a patients receiving radiotherapy. Result of work was the system described and published as a chapter in a monograph [9]. The project aimed at visualization of data, and carry out simple online statistical analyzes. The experience gained while working on this project allowed for analyze the more complex aspects of machine learning. Next research projects were related to clustering and classification issues in microarray data. It was system based on WEKA [10], which was designed to choose appropriate clustering or classification method for provided data. Second project was system for microarray re-annotation [11]. Microarray data consist of gene expressions values taken under different conditions. During the work on gene clustering it has become very clear that the key to success will be appropriate extraction attributes (conditions). Clusters of genes with good quality were obtained only after elimination of irrelevant one. It clearly pointed that those clusters reveals some similarity and are recognizable as a group only on subset of conditions. This raised the issue of bi-clustering.

First attempt for bi-clustering analysis was done in publication describing project of dynamic clustering of document database [12]. System assumes that document provided on the input were translated into word-occurrence matrices. Next on this basis system performs bi-clustering analysis and extract aspects which appears in document. Such information where further used for search proposes. Whole system was based on non-negative matrix factorization algorithms described in chapter 6.1. Last to projects were presented on conferences and was about comparisons of bi-clustering algorithms [13] and about distributed system for running bi-clustering experiments [14]. Those projects will be described later in this work.

5. Formulation of main problems

5.1. Definition of bi-clusters

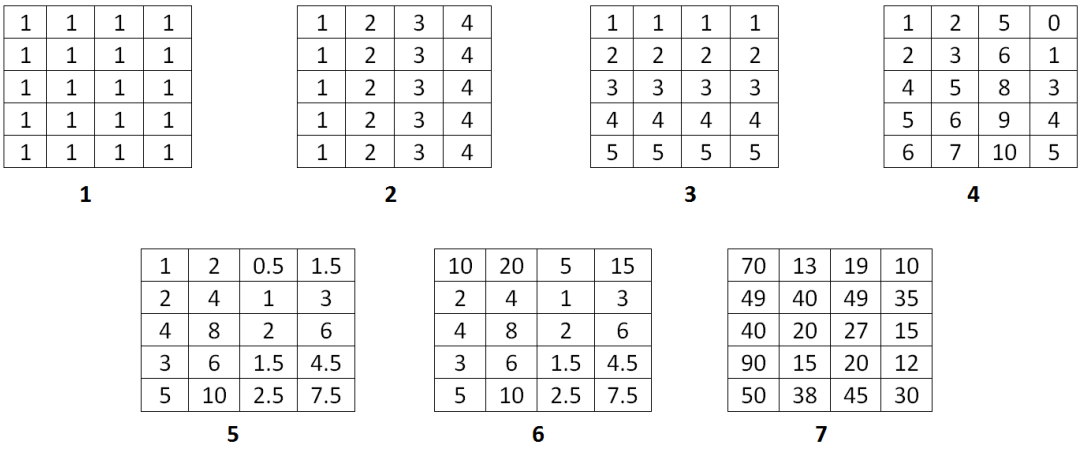


Figure 5. Bi-cluster types: 1) Constant, 2) Constant on columns, 3) Constant on rows, 4) Coherent values (additive model), 5) and 6) Coherent values (multiplicative model) 7) Coherent evolutions

Notation was taken from the paper by Madeira & Oliveira [15], where bi-cluster is defined by a subset of rows and subset of columns from data matrix. Given the data matrix V with set of rows (X), and set of columns (Y), a bi-cluster (B) is defined by a sub-matrix (I, J), where I is a subset of X, and J is a subset of Y.

$$V = (X, Y),$$

$$V = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ a_{M1} & a_{M2} & \cdots & a_{MN} \end{bmatrix}, X_i = [a_{i1} \quad a_{i2} \quad \cdots \quad a_{iN}], Y_i = \begin{bmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{Mj} \end{bmatrix}$$

$$B = (I, J), I \in X, J \in Y$$

Single bi-clustering experiment (R) outputs K bi-clusters, where K is a number which, depending on the algorithm used, can be a parameter given by the user, or the number formed as a result of executing the selected method.

$$R = \{B_i = (I_i, J_i)\} \; where \; i = 1 \ldots K \; and \; \forall i: I_i \in X, J_i \in Y$$

Determining the exact number of clusters in data is a difficult task to perform. Usually user tries a range of values, so that some index function determining the

quality bi-clusters is maximized. Examples of quality indexes are described in the Chapter 5.2.

We distinguish few classes of bi-clusters (Figure 5):

- Bi-clusters with constant values (1). Perfect bi-cluster in this class is the one whose values match the following formula:

$$a_{ij} = \mu$$

  Where:

  - μ – is typical value within bi-cluster

  This is the easiest bi-cluster to find because it can be read directly from data matrix. Algorithms specialized in such task usually group similar rows and columns, splits original matrix into smaller matrices in which it check variance within the bi-cluster. Such approach is able to find cluster with the same value over whole bi-cluster but is not very resistant to noise in the data.

  First attempt to finding constant bi-cluster was "Block Clustering" by Hartigan [2]. He implemented the approach described in the previous paragraph – splitting data matrix into smaller matrices and then computing variance over its elements:

$$VAR(I,J) = \sum_{i \in I, j \in J} \left(a_{ij} - a_{IJ}\right)^2$$

  Where a_{IJ} is an average value in bi-cluster.

  To avoid situation in which algorithm splits data matrix over sub-matrices containing only one element, author add stop criteria for maximum number of bi-cluster:

$$VAR(I,J)_K = \sum_{k=1}^{K} \sum_{i \in I^k, j \in J^k} \left(a_{ij}^k - a_{IJ}^k\right)^2$$

  Where a_{IJ} is an average value in bi-cluster.

  Tibshirani et al. [16] propose another variance based algorithm by modification of Hartigan [2]. His modification was introducing backward pruning method to splitting and permutation based method for finding optimal number of bi-clusters.

  Another worth mentioning algorithm for finding constant bi-clusters is Double Conjugated Clustering by Busygin et al. [17]. Its two way clustering method which

perform simple clustering and next computes similarities between rows and columns, which leads to finding constant bi-clusters.

- Bi-clusters with constant values on rows or columns (3 or 2), Perfect bi-cluster in this class is the one whose values match the following formula:

$$a_{ij} = \mu + \alpha_i \text{ or } a_{ij} = \mu + \beta_j$$

  Where:

  - μ – is typical value within bi-cluster,
  - α_i – is adjustment for row i,
  - β_j – is adjustment for column j.

The task of detecting clusters with constant rows or columns is very similar to the detection of constant clusters. It can be very easily brought to it by normalizing the rows or the columns by row or column mean respectively.

- Bi-clusters with coherent values (additive model) (4). In literature also known as "shift bi-clusters". Perfect matrix with coherent values in additive model follow given expression:

$$a_{ij} = \mu + \alpha_i + \beta_j,$$

  Where:

  - μ – is typical value within bi-cluster,
  - α_i – is adjustment for row i,
  - β_j – is adjustment for column j.

- Bi-clusters with coherent values (multiplicative model) (5,6). In literature also known as "scale bi-clusters". Perfect matrix with coherent values in multiplicative model follow given expression:

$$a_{ij} = \mu * \alpha_i * \beta_j,$$

  Where:

  - μ – is typical value within bi-cluster,
  - α_i – is adjustment for row i,
  - β_j – is adjustment for column j.

- Bi-clusters with coherent evolutions (7). In literature also known as shift-scale bi-clusters. Definitely the most difficult type of clusters to explore. This

point is proven by Kemal Eren [18] in his comparative analysis of bi-clustering algorithms. Bi-clusters with coherent evolutions have made the difficulty for almost each of the analyzed algorithms. Only CPB algorithm [19] perform quite well on such type of data. In Chapter 7 I'm trying to reproduce and extend their results. Formula describing data in bi-clusters with coherent evolutions follow given expression:

$a_{ij} = \mu * \alpha_i * \beta_j + \rho_i + \gamma_j$ (scale-shift model)

$a_{ij} = (\mu + \rho_i + \gamma_j) * \alpha_i * \beta_j$ (shift-scale model)

Where:

- μ – is typical value within bi-cluster,
- α_i – is scaling parameter for row i,
- β_j – is scaling parameter for column j,
- ρ_i – is shifting parameter for row i,
- γ_j – is shifting parameter for column j.

- Plaid model bi-clusters (Not included in Figure 5). In literature also known as General Additive model. Algorithms specialized in this type of bi-clusters can be useful in case of data presented in Figure 6 (plot number 5).Plaid model consist of background layer and series of coherent layers.

$$a_{ij} = (\mu_0 + \alpha_{i0} + \beta_{j0}) + \sum_{k=1}^{K} (\mu_k + \alpha_{ik} + \beta_{jk}) * \delta_{ik} * \omega_{jk}$$

Where:

- μ_0 – is typical value for background layer,
- μ_k – is typical value within bi-cluster k,
- α_0 – is shifting parameter for background,
- β_0 – is shifting parameter for background,
- α_{ik} – is shifting parameter for row i in bi-cluster k,
- β_{jk} – is shifting parameter for column j in bi-cluster k,
- δ_{ik} – is binary indicator of membership i row in bi-cluster k,
- ω_{jk} – is binary indicator of membership j column in bi-cluster k.

Each of the above formulas that describe the data structure of the bi-clusters, relates to ideal case where data do not contain any noise. Unfortunately, real life is not perfect and data without noise does not exist. It should be taken into account in formulas:

$$a_{ij} = a_{ij} + \varepsilon_{ij}$$

Where ε_{ij} is random noise in cell from i row and j column.

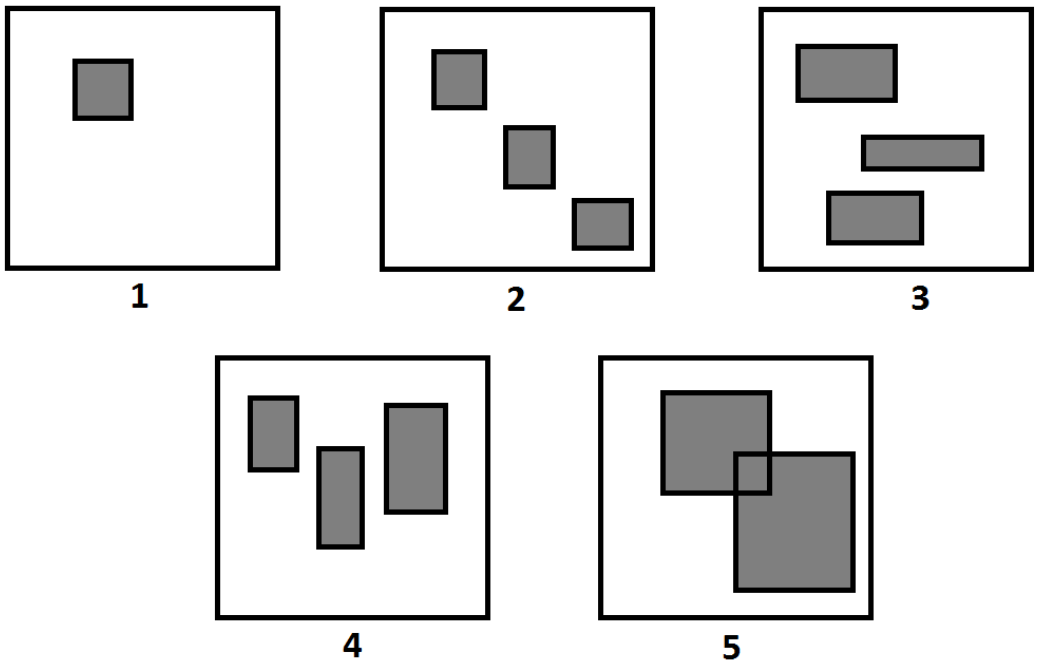


Figure 6. Bi-cluster structures.

Bi-clusters can also be divided according to the structure. Figure 6 shows an example structures. 1) a single cluster, 2) Bi-clusters exclusive on columns and rows. These two types, as a matter of fact, do not require the use of bi-clustering methods. To find them, is sufficient to use classic clustering approach. This is where bi-clustering is most useful is more complex structures such as: 3) overlapping columns, 4) overlapping rows, 5) overlapping in both dimensions.

5.2. Index functions for evaluating quality of bi-clustering systems

5.2.1. Mean square residue (MSR)

This score was proposed by Cheng and Church [4] in 2001. It can be applied to results where bi-cluster structure is known, and it is constant (on whole bi-cluster or only constant columns/rows). If we have subsets I and J, than we can compute residue for each element a_{ij}(single element of matrix indicated by the subsets I and J):

$$a_{ij} - a_{iJ} - a_{Ij} + a_{IJ}$$

Where:

- a_{ij} – value of element in i'th row and j'th column of bicluster,
- a_{iJ} – mean of i'th row,
- a_{Ij} – mean of j'th column,
- a_{IJ} – mean of all elements in the bicluster.

Formula for MSR is defined as follows:

$$H(I,J) = \frac{1}{|I||J|} \sum_{i \in I, j \in J} (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2$$

where

$$a_{iJ} = \frac{1}{|J|} \sum_{j \in J} a_{ij}\,, a_{Ij} = \frac{1}{|I|} \sum_{i \in I} a_{ij}$$

and

$$a_{IJ} = \frac{1}{|I||J|} \sum_{i \in I, j \in J} a_{ij} = \frac{1}{|I|} \sum_{i \in I} a_{iJ} = \frac{1}{|J|} \sum_{j \in J} a_{Ij}$$

The mean square residue is the variance of the set of all elements in the bi-cluster. It should be zero or close to zero in constant bi-cluster, or below certain threshold in general. This method is suitable for bi-clusters with constant values or coherent values with additive model.

5.2.2. Average Correlation Value (ACV)

AVC property was proposed by Li Teng and Laiwn Chan [20] in 2007. Authors assume that bi-cluster should be a subset of attributes from both dimension that are

highly correlated. Based on this assumption AVC value of bi-cluster A is calculated using following formula:

$$\overline{R}(A) = max\left\{\frac{\sum_{i=1}^{n}\sum_{j=1}^{n}|r\_row_{ij}| - n}{n^2 - n}, \frac{\sum_{k=1}^{m}\sum_{l=1}^{m}|r\_col_{kl}| - m}{m^2 - m}\right\}$$

$$\overline{R}(A) \in [0,1]$$

Where:

- r_row_{ij} – is the correlation between the i'th and j'th rows,
- r_col_{kl} – is the correlation between k'th and l'th columns.

Large value of $\overline{R}(A)$ means that rows and columns of bi-cluster A are highly correlated with each other. Measure is suitable for constant, additive and multiplicative bi-clusters.

5.2.3. Average Spearman's rho (ASR)

This measure was proposed by Wassim Ayadi et. al [21] in response to previous measure [20]. Authors introduce change in way how correlation is computed, in order to improve the results. The formula is as follows:

$$ASR(A) = 2 * max\left\{\frac{\sum_{i=1}^{n}\sum_{j=i+1}^{n} p_{ij}}{|I|(|I| - 1)}, \frac{\sum_{k=1}^{m}\sum_{l=k+1}^{m} p_{kl}}{|J|(|J| - 1)}\right\}$$

$$-1 \leq ASR(A) \leq 1$$

Where p_{ij} is *Spearman's rank correlation*, and it's used to expresses the correlation between two vectors (i.e. $X_i = \left(x_1^i, x_2^i, \dots, x_m^i\right)$ and $X_j = \left(x_1^j, x_2^j, \dots, x_m^j\right)$) is defined as follows:

$$p_{ij} = 1 - \frac{6\sum_{k=1}^{m}\left(r_k^i\left(x_k^i\right) - r_k^j\left(x_k^j\right)\right)^2}{m(m^2 - 1)}$$

Where $r_k^i\left(x_k^i\right)$ (resp. $r_k^j\left(x_k^j\right)$) is the rank of x_k^i (resp. x_k^j).

Measure is suitable for bi-clusters of any type (Figure 5). And it results value of 1, which indicates that attributes within bi-cluster are highly correlated, and value of -1 if very weakly.

The following table shows how these measures are useful depending on the type of bi-cluster (from Figure 5). As there is clearly see MSR measure is good only for constant bi-clusters or additive bi-clusters. ACV and ASR measures are suitable for all types of bi-clusters, but ASR is a little bit better than ACV in case of bi-clusters with coherent evolutions.

Table 1. Comparison of evaluation functions on bi-clusters from Figure 1.

Bi-cluster / Function	Ex-pected value	1	1	2	3	4	5	6	7
MSR	0	0	0	0	0	0	0.62	2.425	131.87
ACV	1	1	1	1	1	1	1	1	0.84
ASR	1	1	1	1	1	1	1	1	0.99

5.3. Stop criteria for bi-clustering algorithms

There are many different methods with different approaches to the bi-clustering task. Some of them are described in Chapter 6. After selecting the appropriate method and setting the parameters, the final decision of the user, prior to starting the experiment, is to decide how and when it is going to end. There is no problem if the chosen method is based on an exhaustive enumeration of columns and rows. Condition of end in them is natural and user cannot change it. But there are many methods where the end condition must be defined. Since the theoretical assumptions imply an infinite number of steps. Below is presented the most popular approaches to this issue.

5.3.1. Mathematical convergence

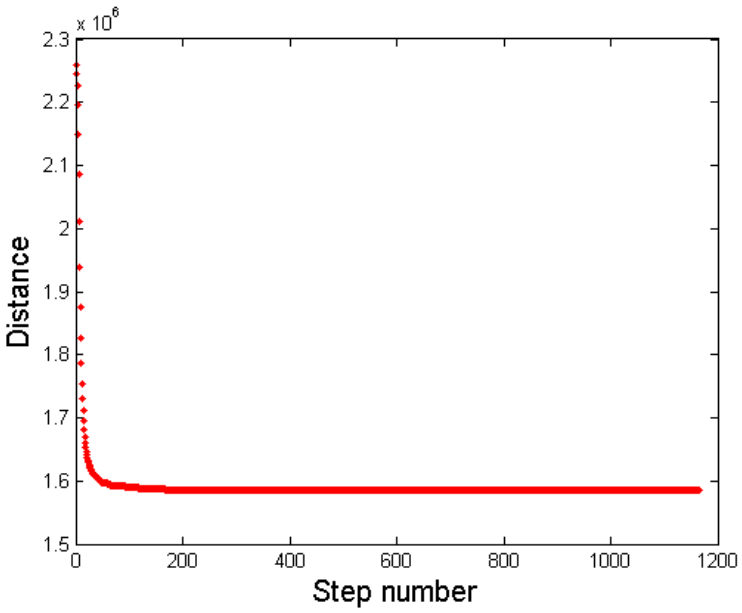


Figure 7. Sample function of change in distance function vs step number.

Convergence is the most natural and intuitive condition for the end of the update rule-based algorithms. Good example of such algorithms is those described in paper written by Sung and Lee [8]. Authors proposed two methods based on multiplicative update rules for minimizing distance functions which represent how data matrix differs from factor matrices. Their idea for bi-clustering is to provide a matrix factorization of data matrix A to product of factor matrices W and H. Bi-clusters are

extracted from factor matrices. Generally this approach of determining the end of the analysis is applicable for all methods based on distance or divergence functions.

$$A = WH$$

First step of this algorithm is initialize matrices with random values. Then proper update rules are designed to minimize the distance from the factors to data matrix. Regarding to assumed distance function.

The only drawback of this approach of determining the end of the analysis is time complexity. Like it is presented on Figure 7 the rate of change of the distance function in subsequent steps decreases very quickly but theoretically never reach zero. In real life this rate is limited by computer precision, but reaching it is impractical due to long time needed. As there is clearly shown in the attached picture, after a certain number of steps the change is imperceptible.

5.3.2. Connectivity matrix

In the case of methods based on the decomposition of the matrix [8], more interested for as is order of the values than the exact values. For example in non-negative matrix factorization algorithms described above single bi-cluster is obtain using single row vector from matrix W and single column vector from matrix H. Multiplication of those two vectors represent data matrix only with one selected bi-cluster (Figure 8).

$$[0\,0\,0\,1\,2\,3\,4\,0\,0\,0] * \begin{bmatrix}0\\0\\0\\1\\2\\3\\4\\0\\0\\0\end{bmatrix} = \begin{bmatrix}0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&0\\0&0&0&1&2&3&4&0&0&0\\0&0&0&2&4&6&8&0&0&0\\0&0&0&3&6&9&12&0&0&0\\0&0&0&4&8&12&16&0&0&0\\0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0&0&0\end{bmatrix}$$

Figure 8. Bi-cluster extraction in NMF algorithms.

In the ideal case non-zero components of the first vector represent the rows of the data matrix involved in the bi-cluster. Subsequent non-zero components of the second vector represent the columns that are involved. Life is unfortunately not perfect and due to noise in both inside and outside of the bi-cluster, very often all values are not zero. But the attributes that actually take part in the resulting cluster should have significantly higher values that those attributes not involved.

The way to select only the relevant attributes can be normalization and to determining the cut-off threshold. One type of such a threshold is the threshold of the first n-elements. With such a defined task, we can assume that the order of the attributes is important more than exact values. The order we determined using exact values. The greatest value is in the first place, the smallest on the last. Can then be assume that if within specific number of steps the order of attributes in all the vectors does not change, we achieved convergence.

$$\underset{Step\ 1}{\begin{bmatrix}1\\3\\4\\2\end{bmatrix}} \Rightarrow \underset{Step\ 2}{\begin{bmatrix}2\\3\\1\\4\end{bmatrix}} \Rightarrow \underset{Step\ 3}{\begin{bmatrix}4\\2\\3\\1\end{bmatrix}} \Rightarrow \cdots \Rightarrow \underset{Step\ n-2}{\begin{bmatrix}4\\1\\2\\3\end{bmatrix}} \Rightarrow \underset{Step\ n-1}{\begin{bmatrix}4\\1\\2\\3\end{bmatrix}} \Rightarrow \underset{Step\ n}{\begin{bmatrix}4\\1\\2\\3\end{bmatrix}}$$

The above example illustrates the way in which convergence is determined. After the first step, the sample vector of the analyzed matrix contains attributes organized in the following way:

- attribute in the first place has a rank 1 because it has the greatest value.
- attribute in the fourth place has a rank 2 because its value its less than attribute with rank 1 and greater from the rest.
- attribute in the second place has a rank 3 because its value is grater only from one attribute.
- attribute in the third place has rank 4 because it has the lowest value.

This order is calculated after each step. If he does not change for a specified number of steps for the calculation are considered to be terminated.

5.3.3. Conditions defined by the user.

The approaches described above, despite the fact that reducing the time from the infinite to the finite, have one drawback. Namely, although the exact definition of the conditions, it is impossible to clearly determine how long it will take to finish the experiment. In order to prevent too long waiting time the user typically specifies one or more of the following conditions:

- maximum number of steps,
- maximum duration of the experiment,
- minimum value of the bi-clusters quality function.

6. An overview of bi-clustering methods

6.1. Algorithms based on matrix decomposition

A very wide range of algorithms are algorithms based on data matrix decomposition. In such methods data matrix (A) is factorized into (usually) much smaller matrices. Such a distribution, because of the much smaller matrices is much easier to analyze, and the obtained matrices reveal previously hidden features. These algorithms are often called NMF algorithms. NMF stands for non-negative matrix factorization. Two efficient algorithms were introduced by Seung and Lee [8]. First minimize conventional least square error distance function and second generalized Kullback–Leibler divergence. Third and last from this group is algorithm that slightly modify the second approach. Author [22] introduce smoothing matrix for achieving a high degree of sparseness, and better interpretability of the results. Data matrix in this techniques is factorized into (usually) two smaller matrices:

$$A \approx WH$$

Finding the exact solution is computationally very difficult task. Instead, the existing solutions focus on finding local extrema of the function describing the fit of the model to the data. Below some examples of such divergence functions.

6.1.1. Based on LSE.

Distance function:

$$\|V - WH\|^2 = \sum_{ij} (V_{ij} - (WH)_{ij})2$$

Update rules:

$$H_{ij} = H_{ij} \frac{(W^T V)_{ij}}{(W^T WH)_{ij}}$$

$$W_{ij} = W_{ij} \frac{(VH^T)_{ij}}{(WHH^T)_{ij}}$$

6.1.2. Based on Kullback–Leibler divergence

Divergence function:

$$D(V||WH) = \sum_{ij}\left(V_{ij}\log\frac{V_{ij}}{WH_{ij}} - V_{ij} + WH_{ij}\right)$$

Update rules:

$$H_{ij} = H_{ij}\frac{\sum_k W_{ki}V_{kj}/(WH)_{kj}}{\sum_l W_{li}}$$

$$W_{ij} = W_{ij}\frac{\sum_k H_{jk}V_{ik}/(WH)_{ik}}{\sum_l H_{jl}}$$

6.1.3. Based on non-smooth Kullback–Leibler divergence.

Divergence function:

$$D(V||WSH) = \sum_{ij}\left(V_{ij}\log\frac{V_{ij}}{WSH_{ij}} - V_{ij} + WSH_{ij}\right)$$

Update rules for this method is the same as in previews one, but instead W in update rule for H we substitute WS, and in update rule for W we substitute SH. Smoothing matrix S looks as follows:

$$S = (1-\theta)I + \frac{\theta}{q}11^T$$

Where:

I – Identity matrix, 1 – vector of ones and θ – should meet condition $0 \leq \theta \leq 1$.

Another type of group NMF algorithms are algorithms based on the expectation-maximization method. Because of the approach, the distance function replaces the likelihood function. Below the examples of such methods.

6.1.4. PLSA

PLSA stands for Probabilistic Latent Semantic Analysis. Introduced by Thomas Hoffman [1], and based on maximizing log-likelihood function. For this purpose author use Expectation-Maximization algorithm [5]. Formulas for computing results:

Log-likelihood function:

$$E[L^c] = \sum_{i=1}^{N}\sum_{j=1}^{M} n(d_i, w_j) \sum_{k=1}^{K} P(z_k|d_i, w_j) \log\left[P(w_j|z_k)P(z_k|d_i)\right]$$

E-step:

$$P(z_k|d_i, w_j) = \frac{P(w_j, z_k)P(z_k, d_i)}{\sum_{l=1}^{K} P(w_j, z_l)P(z_l, d_i)}$$

M-step:

$$P(w_j|z_k) = \frac{\sum_{i=1}^{N} n(d_i, w_j)P(z_k|d_i, w_j)}{\sum_{m=1}^{M}\sum_{i=1}^{N} n(d_i, w_m)P(z_k|d_i, w_m)}$$

$$P(z_k|d_i) = \frac{\sum_{i=1}^{N} n(d_i, w_j)P(z_k|d_i, w_j)}{n(d_i)}$$

The author explains the meaning of those formulas by using the example. Factor w_j represent one word from vocabulary that contains M words. Factor d_i represents one of N documents. And z_k means aspect. Expression $n(d_i)$ denotes number of words in document i, and $n(d_i, w_j)$ denotes number of occurrences of word j in document i.

Translating the data generation process into a joint probability model results in the expression:

$$P(w_j|d_i) = \sum_{k=1}^{K} P(w_j|z_k)P(z_k|d_i)$$

In above equation all possible probabilities $P(w_j|d_i)$ form a data matrix (in our notation V) with M rows and N columns. Authors assume that this matrix contains K bi-clusters. Data matrix is factorized into two smaller matrices. The first one

has M rows and K columns, and represents the probability of occurrence of a word in the context of aspect. The second consists of K rows and N columns, and represents probability of an aspect in the document. Single bi-cluster is in the matrix formed from the product of k-th column from first matrix and k-th row.

6.1.5. FABIA

FABIA stands for Factor Analysis for BIClustering Acquisition. Algorithm were introduced by Hochreiter [23] and based on Expectation-Maximization algorithm.

E-step

$E(z_j|x_j) = (\Lambda^T \Psi^{-1} \Lambda + \Xi_j^{-1})^{-1} \Lambda^T \Psi^{-1} x_j$ and

$$E(z_j z_j^T|x_j) = (\Lambda^T \Psi^{-1} \Lambda + \Xi_j^{-1})^{-1} + E(z_j|x_j) E(z_j|x_j)^T$$

Where Ξ_j stands for $diag(\xi_j)$, where update for ξ_j is:

$$\xi_j = diag(\sqrt{E(z_j z_j^T|x_j)})$$

M-step:

$$\Lambda^{new} = \frac{\frac{1}{l}\sum_{j=1}^{l} x_j E(z_j|x_j)^T - \frac{\alpha}{l}\Psi sign(\Lambda)}{\frac{1}{l}\sum_{j=1}^{l} E(z_j, z_j^T|x_j)}$$

$$diag(\Psi^{new}) = diag\left(\frac{1}{l}\sum_{j=1}^{l} x_j x_j^T - \Lambda^{new}\frac{1}{l}\sum_{j=1}^{l} E(z_j|x_j) x_j^T\right) + diag\left(\frac{\alpha}{l}\Psi sign(\Lambda)\ (\Lambda^{new})^T\right)$$

Where:

- z – vector of factors,
- x – sample from data matrix,
- Λ – sparse prototype matrix,
- Ψ- covariance matrix – expressing independent noise,

- ξ – variational parameter,
- l – number of factors.

Data initialization:

1) vectors ξ_j by ones

2) Λ randomly

3) $\Psi = diag(\max(\delta, covar(x) - \Lambda\Lambda^T))$

Model likelihood is define as follows:

$$p(x|\Lambda, \Psi) = \int p(x|z, \Lambda, \Psi)p(z)\, dz$$

Where:

$$p(z) = \left(\frac{1}{\sqrt{2}}\right)^p \prod_{i=1}^{p} e^{-\sqrt{2}|z_i|}$$

Likelihood function introduce a model family that is parameterized by ξ, where the maximum over models in this family is the true likelihood:

$$arg \max_{\xi} p(x|\xi) = p(x)$$

6.2. Algorithms based on bipartite graphs

6.2.1. QUBIC

QUBIC stands for QUalitative BIClustering algorithm. It was proposed by Guojun Li, et al. [5] as very efficient algorithm for analysis of gene expression data. Authors proposed weighted graph representation of discretized expression data. The expression levels are discretized to the ranks. Their number is determined by the user through the parameters of the algorithm. Number of ranks is essential and strongly affects the results. The algorithm allows two types of ranks. The positive (for up-regulating genes) and negative sign (for down-regulating genes). The vertices of the graph represent genes. The edges between them have weight to reflect the number of conditions for which they have the same rank.

Algorithm starts with translating data matrix into new representation, which is a graph where vertex set is built from rows. An intermediate step is to create a matrix of integers. This matrix is the same size as original data matrix and its values are created as follows:

1. For each row i all values are sorted in increasing order:

   $$a_{i1} \dots a_{i,s-1} a_{is} \dots a_{i,c-1} a_{i,c} a_{i,c+1} \dots a_{i,m-s+1} a_{i,m-s+2} \dots a_{im}$$

   Where:

   m – number of columns

   $c = \frac{m}{2}$ – the median value in a row

   $s = m * q + 1$ – number which determine how many values will be marked as zero. q is parameter selected by the user
2. Values are marked as zero if a_{ij} belongs to interval $(a_{ic} - d_i, a_{ic} + d_i)$ where $d_i = \min(a_{ic} - a_{is}, a_{i,m-s+1} - a_{ic})$
3. Values are marked with positive ranks from range <1 , r> if $a_{ij} > a_{ic} + d_i$
4. Values are marked with positive ranks from range <1 , r> if $a_{ij} < a_{ic} - d_i$

$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array}\begin{bmatrix} -1 & 2 & -2 & 0 & 1 \\ 1 & 0 & -2 & 2 & -1 \\ -1 & 2 & 0 & -2 & 1 \\ 1 & -1 & 0 & 2 & -2 \\ 2 & -1 & 1 & 0 & -2 \end{bmatrix}$$

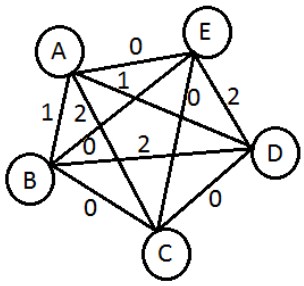


Figure 9. Sample QUBIC transformation from matrix of integers to final graph.

Bi-clusters are find one-by-one. Starting from single heaviest and unused edge as seed, algorithm iteratively add additional edges until its violates pre-specified consistency level.

6.3. Algorithms based on Iterative Row and Column search

6.3.1. Coupled Two-Way Clustering (CTWC)

CTWC is a bi-clustering technique propose by Gad Getz et. al [7] in 2000. They deal with gene expression data from microarray experiments. The purpose of their work was to developed algorithm for identifying biologically relevant partitions in data using unsupervised learning.

Authors present their work using gene expression matrices. Values in such data matrix represents expression value of a gene measured on some sample. Authors use following notation:

g – set of genes

s – set of samples

First step of an algorithm is to perform standard two-way clustering on whole data matrix. It means that we start with g^0 and s^0 which represents respectively whole set of genes and whole set of samples. The results of such will be clusters g_i^1 and s_j^1, which are respectively subsets of genes and subsets of samples.

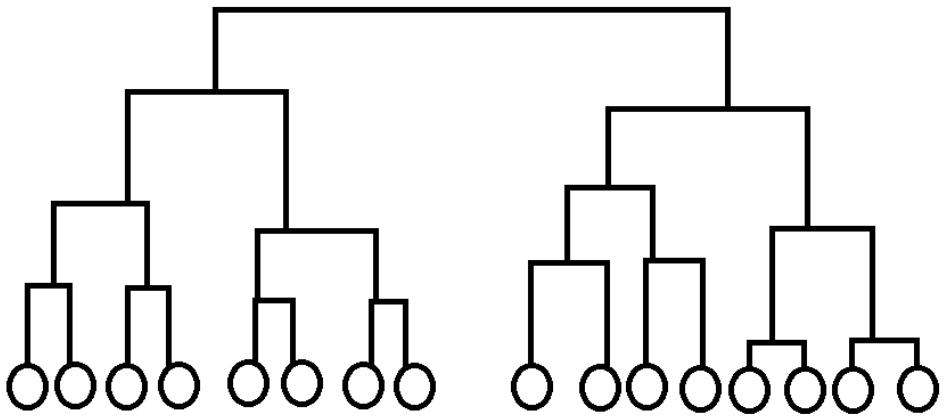


Figure 10. Example of hierarchical clustering.

Next for every step k two-way clustering is performed between every pair of clusters (g_i^n, s_j^m) where n and m are from range 0 to k-1. Result after this step will be cluster denoted as g_l^k and s_h^k. Such process is visual

6.4. Algorithms based on Divide and Conquer approach

6.4.1. Block clustering

In 1972 Hartigan [2] proposed an algorithm known as “Block Clustering”. The idea is based on splitting original data matrix into sub-matrices and looking for those with smaller variance:

$$VAR(I,J) = \sum_{i \in I, j \in J} \left(a_{ij} - a_{IJ}\right)^2$$

Where a_{IJ} is bi-cluster mean value.

Such define measure is designed for finding constant bi-clusters, because those have variances equal or close to zero. But also such a measure obviously likely to favor bi-clusters composed of only one column and one row. To avoid this, one of the input parameters, is the maximum number of clusters that we want to find. Due to the quality measure algorithm looks only for bi-clusters with constant values, but the author mentions about modifications in merit function which make it possible to find bi-cluster with constant row or columns or even coherent values. The idea of block pruning, proposed by Hartigan is visualized on Figure 11.

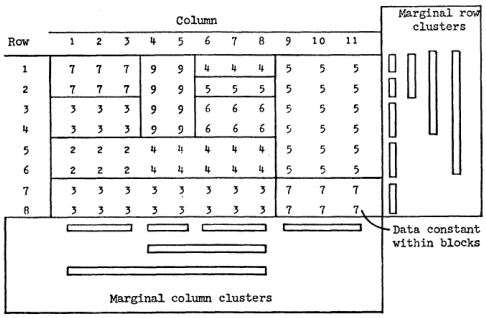


Figure 11. Example of block clustering. Figure taken from original Hartigan publication [2].

In 1999 Tibshirani et al. [16] propose modifications in Hartigan method, which allows to induce the number of bi-clusters. Modifications were backward pruning method for block splitting, and permutation-based method for finding optimal number of clusters. However, the merit function remain the same, so algorithms is still for constant bi-cluster only.

6.5. Algorithms based on Greedy iterative search

6.5.1. δ-bi-clusters

This algorithms is commonly referred by the names of its authors, Cheng and Church [4]. Authors in 2001 applied as first bi-clustering to microarray data. Their method still remains as important benchmark to every new data and method. Proposed approach is based on measure how elements differs from row mean, column mean and overall mean:

$$H(I,J) = \frac{1}{|I||J|} \sum_{i \in I, j \in J} (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2$$

where

$$a_{iJ} = \frac{1}{|J|} \sum_{j \in J} a_{ij} \, , a_{Ij} = \frac{1}{|I|} \sum_{i \in I} a_{ij}$$

and

$$a_{IJ} = \frac{1}{|I||J|} \sum_{i \in I, j \in J} a_{ij} = \frac{1}{|I|} \sum_{i \in I} a_{iJ} = \frac{1}{|J|} \sum_{j \in J} a_{Ij}$$

Method aims at finding largest bi-clusters with respect to $H(I,J)$, which shouldn't be above threshold δ. Algorithm start with largest possible bi-cluster and consistently removes columns and rows as long as the value of the quality function is below a certain level δ. Below algorithm for deleting rows or columns:

Algorithm: node deletion
input: matrix A , row set I , column set J , $\delta >= 0$
output: row set I' and column set J' so that $H(I',J') <= \delta$
while $H(I,J) > \delta$:
    find the row $r = \max_{i \in I} d(i)$ and the column $c = \max_{j \in J} d(j)$
    if $d(r) > d(c)$ **then** remove r from I else remove c from J
return I and J

Where:

$$d(i) = \frac{1}{|J|} \sum_{j \in J} \left(a_{ij} - a_{iJ} - a_{Ij} + a_{IJ}\right)^2 \; and \; d(j) = \frac{1}{|I|} \sum_{i \in I} \left(a_{ij} - a_{iJ} - a_{Ij} + a_{IJ}\right)^2$$

6.6. Algorithms based on Exhaustive bi-cluster enumeration

6.6.1. Statistical-Algorithmic Method for Bi-cluster Analysis (SAMBA)

Algorithm is based on translating data into join probability model to identify subset of row that jointly respond across a subset of columns in data matrix [24]. Original data is modeled as bi-partite graph where rows and columns are respectively two of its set. Vertex are weighted accordingly to probabilistic model, and bi-clusters appears as heavy sub-graphs. Result bi-clusters are obtain by heuristic search, and reducing vertices.

SAMBA model assume that data is represented as bi-partite graph $G = (U, V, E)$ where U is a set of columns, V is a set of rows and E is a set of edges between them. Bi-clusters in such approach are represented by heavy sub-graphs $B = (U', V', E')$ where U' and V' are respectively subset of columns that reveals some similarity on a subset of rows. Method assumes that bi-clusters represent approximately uniform relations between their elements. It leads to conclusion that each edge of a bi-cluster occurs with constant high probability p_c. The log likelihood for B is define as follows:

$$\log L(B) = \sum_{(u,v)\in E'} \log \frac{p_c}{p_{u,v}} + \sum_{(u,v)\in \overline{E'}} \log \frac{1-p_c}{1-p_{u,v}}$$

Where $\overline{E'} = (U' \times V') \backslash E'$

6.7. Algorithms based on Distribution parameter identification

6.7.1. Plaid Model

Plaid model is modeling method proposed by Lazzeroni and Owen [25]. Approach is based on statistics and authors applies it to gene expression analysis. The key idea is to represent original matrix as a superposition of layers, which should correspond to bi-clusters.

Model assumes that data matrix is a sum of uniform background and k bi-clusters. Its described by following equation:

$$a_{ij} = \mu_0 + \sum_{k=1}^{K} (\mu_k + \alpha_{ik} + \beta_{jk}) * \delta_{ik} * \omega_{jk}$$

Where:

- μ_0 – is typical value for background layer,
- μ_k – is typical value within bi-cluster k,
- α_{ik} – is shifting parameter for row i in bi-cluster k,
- β_{jk} – is shifting parameter for column j in bi-cluster k,
- δ_{ik} – is binary indicator of membership i row in bi-cluster k,
- ω_{jk} – is binary indicator of membership j column in bi-cluster k.

Authors formulate problem as minimization of distance function between model and original data:

$$\sum_{ij} \left(A_{ij} + \sum_{k=1}^{K} \theta_{ijk} * \delta_{ik} * \omega_{jk} \right)^2$$

Lazzeroni and Owen propose an iterative heuristic to solve this problem of estimating parameters. At every single iteration only one layer is added.

7. Comparing the results

7.1. Similarity measures

7.1.1. Jaccard Index

The easiest way to compare the two sets A and B is the Jaccard index ($\frac{A \cap B}{A \cup B}$). It provides a score of 1 if the sets are identical, and 0 if they are mutually exclusive. So defined index can be used to compare bi-clusters, if we take its constituent clusters individually. If a single bi-cluster $B = (I, J),\ where\ I \in X, J \in Y$ treat as a set consisting of the set of I and J, we can compute average Jaccard index over both clusters.

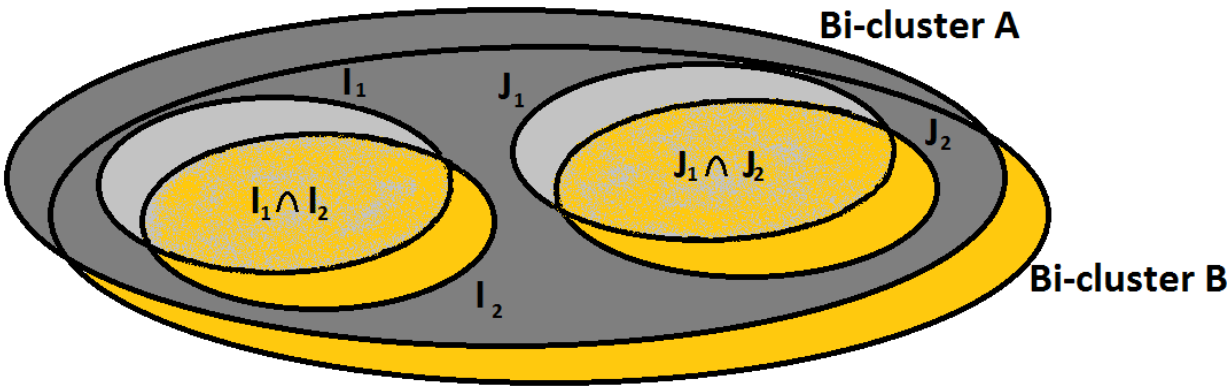


Figure 12. Graphical representation of bi-cluster similarity.

$$S_{Jacc}(B_1, B_2) = \frac{\frac{I_1 \cap I_2}{I_1 \cup I_2} + \frac{J_1 \cap J_2}{J_1 \cup J_2}}{2}$$

But if we do not want to lose the differences arising from the size of individual clusters, then we can use a weighted average:

$$S_{Jacc\_weight}(B_1, B_2) = \frac{(\overline{I_1} + \overline{I_2})\frac{I_1 \cap I_2}{I_1 \cup I_2} + (\overline{J_1} + \overline{J_2})\frac{J_1 \cap J_2}{J_1 \cup J_2}}{\overline{I_1} + \overline{I_2} + \overline{J_1} + \overline{J_2}}$$

Where $B_1 = (I_1, J_1)\ and\ B_2 = (I_2, J_2)$

7.1.2. Relevance and recovery

During comparing the obtain results with the expected one significant are two pieces of information:

- Did we found all expected bi-clusters?
- Did all founded bi-clusters were expected?

Measure described first one is called recovery and second one is relevance. It can be computed using the same formula:

$$S_R(R_1, R_2) = \frac{1}{|R_1|} \sum_{B_1 \in R_1} \max_{B_2 \in R_2} S_{Jacc}(B_1, B_2)$$

Where:

- R_1, R_2- are two set of bi-clusters, coming from different experiments or expected and resulted set
- B_1, B_2-are single bi-clusters coming respectively from R_1, R_2

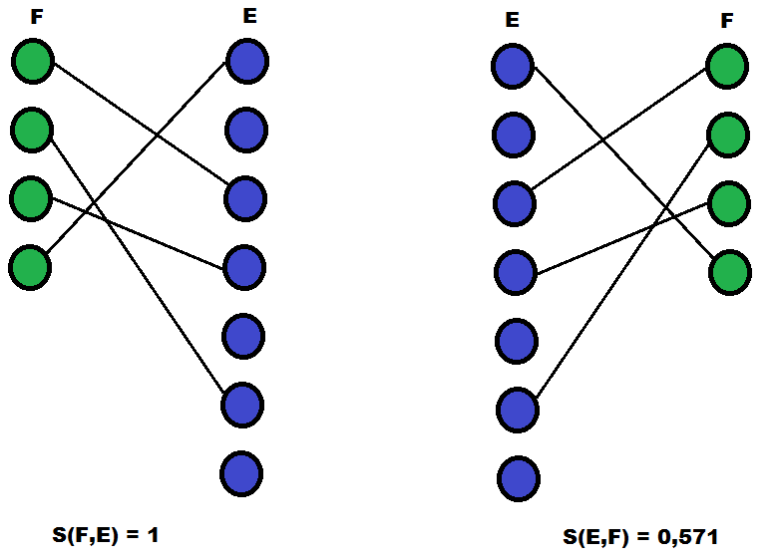


Figure 13. Differences between relevance and recovery.

Similarity function S_R measure how result R_1 fits result R_2. Figure 13 shows a simple example of how to interpret the results. Suppose that there are two sets of bi-clusters. First (blue, marked with letter "E") known in advance and describing the expected results. The second one (green, marked with letter "F") is a set of bi-clusters derived from the analysis. In the ideal case, the two sets should be identical.

In the example set obtained in experiment does not contain all desired bi-clusters. For reasons of simplification it is assumed that the bi-clusters that were obtained contains exact equivalents in the set expected (Jaccard index between connected bi-clusters is equal one). If we check how the "founded set" fit the expected $S_R(F, E)$ it will be called relevance, because it check did all founded bi-clusters are expected. If we approach the task from the other side, that is, if we check how expected set fits founded bi-clusters $S_R(E, F)$ it will be called relevance. It is desirable that both of these measures are equal to one.

7.1.3. Consensus score

Jaccard Index can be applied to comparison of single bi-clusters. When combined with the Hungarian algorithm (Munkres algorithm - described in more detail in Chapter 5.2) can be expanded to use for comparing different results or methods. This quality index called by author "consensus score" was proposed by S. Hochreiter et al. 2010 [23]. Algorithm is as follows:

- Compute similarities between obtained bi-clusters and known bi-clusters from original set (assuming that the bi-clusters are known), or similarities between clusters from first and second result sets.
- Using Munkers algorithm assign bi-clusters of the one set to the bi-clusters from the other one.
- Divide the sum of similarities of the assigned bi-clusters as emphasized number of bi-clusters of the larger set.

Such approach finds assignments witch maximize following function S:

$$S(R_1, R_2) = \sum_{l=1}^{K} S_{Jacc}(B_l^1, B_{l'}^2)$$

Where R_1 *and* R_2 are two independent bi-clustering experiments and B_l^1 *and* $B_{l'}^2$ are pairs of bi-clusters such that B_l^1 is $l'th$ bi-cluster from result R_1 and $B_{l'}^2$ is bi-cluster corresponding to it from result R_2.

As a similarity index Jaccard index S_{Jacc} is used, and if outcome of function S is divided by number of bi-cluster (K) the similarity of two results expressed in percentages is obtained:

$$0 \leq \frac{S(R_1, R_2)}{K} \leq 1$$

A single experiment gets the value 1 if the received bi-clusters are the same as expected, and the value 0 if they are completely different.

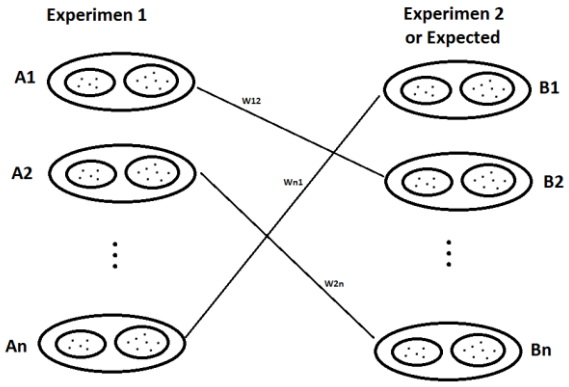


Figure 14. Consensus score algorithm shown by bipartite graph.

This process can be also consider as a bipartite graph analysis. If so, two groups of vertices will be represented by two sets of bi-clusters (from two experiments or the experiment and expected set). Initially, each two vertices from different groups are connected by an edge. Each edge is described by weight, which determines the similarity (S_{Jacc_weight} or S_{Jacc}) between two bi-clusters (vertices). After the Hungarian algorithm, remains only those edges that form a unique pairs of bi-clusters between sets, and its weights form the largest sum.

7.2. Hungarian algorithm

The algorithm was developed and published by Harold Kuhn [26] in 1955, who gave the name "Hungarian algorithm" because the algorithm was based on the earlier works of two Hungarian mathematicians: Dénes Kőnig [27] and Jenő Egerváry [28]. Munkres [29] reviewed the algorithm in 1957 and observed that it is indeed polytime. Since then the algorithm is also known as Kuhn-Munkres algorithm. Although the Hungarian contains the basic idea of the primal-dual method, it solves the maximum weight bipartite matching problem directly without using any linear programming (LP) machinery. Algorithm is based on König's theorem (1916):

If the elements of a matrix are divided into two classes by a property R, than the minimum number of lines that contain all the elements with the property R is equal to the maximum number of elements with the property R, with no two elements on the same line.

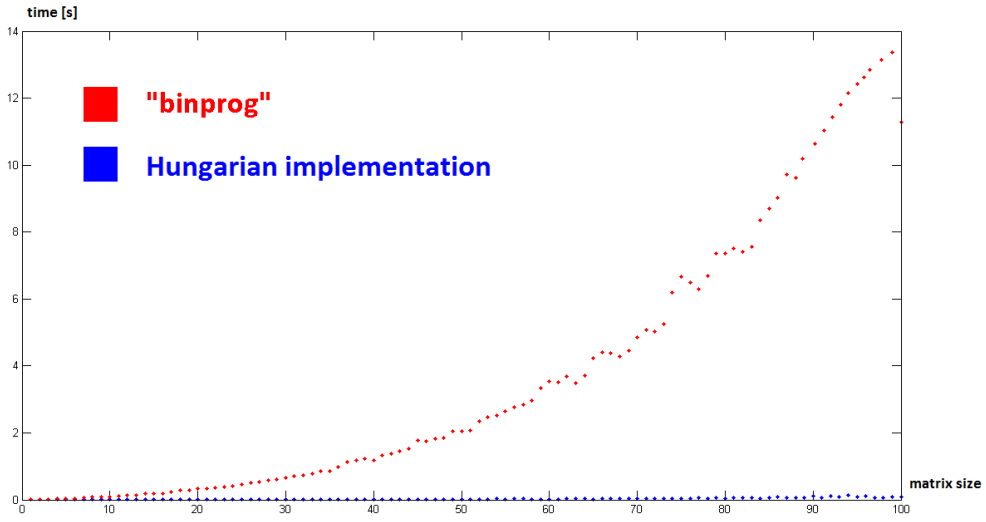


Figure 15. Comparison between Munkres algorithm and classical linear programming approach.

This algorithm is widely used for solving assignment problems in two-dimensional data because of its simplicity and speed. Figure 15 shows comparison between time consumption of Munkres algorithm and classical linear programming algorithm. It has been choosen matlab build-in function “binprog” witch solves binary integer programming problem, and implementation of Hungarian algorithm by Alexander Melin downloaded from MathWorks web site. As it is clearly see in the

attached picture Hungarian algorithm is much faster than the traditional approach, and it is characterized by linear complexity.

Pseudo_code algorithm is as follows:

- Step 1: For each row, subtract the minimum number in that row from all numbers in that row.
- Step 2: For each column, subtract the minimum number in that column from all numbers in that column.
- Step 3: Draw the minimum number of lines to cover all zeroes. If this number = *m*, STOP — an assignment can be made.
- Step 4: Determine the minimum uncovered number (call it *Θ*).
    - Subtract *Θ* from uncovered numbers.
    - Add *d* to numbers covered by two lines.
    - Numbers covered by one line remain the same.
    - Then, GO TO STEP 3.

And pseudo code for resolving problem in Step 3:

- Finding the Minimum Number of Lines and Determining the Optimal Solution
    - Step 1: Find a row or column with only one unlined zero and circle it. (If all rows/columns have two or more unlined zeroes choose an arbitrary zero.)
    - Step 2: If the circle is in a row with one zero, draw a line through its column. If the circle is in a column with one zero, draw a line through its row. One approach, when all rows and columns have two or more zeroes, is to draw a line through one with the most zeroes, breaking ties arbitrarily.
    - Step 3: Repeat step 2 until all circles are lined. If this minimum number of lines equals *m*, the circles provide the optimal assignment.

Example:

Let's consider task in which we have to assign four workers to four jobs. Each job can be perform only by one worker, and each worker can perform only one job. In addition cost of final assignment should be minimal. In classical linear programming approach this task leads to minimization of following function:

$$\min_{ij} \sum \sum c_{ij} x_{ij}$$

Under following conditions:

$$\sum_{j} x_{ij} = 1 \; for \; each \; row,$$

$$\sum_{i} x_{ij} = 1 \; for \; each \; column,$$

$$x_{ij} = 0 \; or \; 1 \; for \; all \; i \; and \; j.$$

Where x_{ij} is an element of binary matrix representing assignments (contains 1 if worker is assign to the job or 0 id not).

Table 2. Example assignment task.

	Job 1	Job 2	Job 3	Job 4
Worker 1	20	22	14	24
Worker 2	20	19	12	20
Worker 3	13	10	18	16
Worker 4	22	23	9	28

In linear programming this problem can be represented by following system of equations:

Min z = $20x_{11} + 22x_{12} + 14x_{13} + 24x_{14} + 20x_{21} + 19x_{22} + 12x_{23} + 20x_{24} + 13x_{31} + 10x_{32} + 18x_{33} + 16x_{34} + 22x_{41} + 23x_{42} + 9x_{43} + 28x_{44}$

s.t.

$x_{11} + x_{12} + x_{13} + x_{14} = 1$ (row 1)

$x_{21} + x_{22} + x_{23} + x_{24} = 1$ (row 2)

$x_{31} + x_{32} + x_{33} + x_{34} = 1$ (row 3)

$x_{41} + x_{42} + x_{43} + x_{44} = 1$ (row 4)

$x_{11} + x_{21} + x_{31} + x_{41} = 1$ (column 1)

$x_{12} + x_{22} + x_{32} + x_{42} = 1$ (column 2)

$x_{13} + x_{23} + x_{33} + x_{43} = 1$ (column 3)

$x_{14} + x_{24} + x_{34} + x_{44} = 1$ (column 4)

x_{ij} >= 0 for i = 1, 2, 3, 4 and j = 1, 2, 3, 4 (nonnegativity)

Solving that equations leads to solution: $x_{11} = 1$, $x_{24} = 1$, $x_{32} = 1$, $x_{43} = 1$

Hungarian algorithm changes a little the function that minimizes:

$$c'_{ij} = c_{ij} - (u_i + v_j) \geq 0$$

$$Maximize \sum_{i=1}^{m} u_i + \sum_{j=1}^{m} v_j$$

So back to our example:

Step 1: Find minimum values in rows and subtract it within each row.

	Job 1	**Job 2**	**Job 3**	**Job 4**
Worker 1	**20**	**22**	**14**	**24**
Worker 2	**20**	**19**	**12**	**20**
Worker 3	**13**	**10**	**18**	**16**
Worker 4	**22**	**23**	**9**	**28**

	Job 1	**Job 2**	**Job 3**	**Job 4**
Worker 1	**6**	**8**	**0**	**10**
Worker 2	**8**	**7**	**0**	**8**
Worker 3	**3**	**0**	**8**	**6**
Worker 4	**13**	**14**	**0**	**19**

Step 2: Find minimum values in columns and subtract it within each column.

	Job 1	Job 2	Job 3	Job 4
Worker 1	6	8	0	10
Worker 2	8	7	0	8
Worker 3	3	0	8	6
Worker 4	13	14	0	19

	Job 1	Job 2	Job 3	Job 4
Worker 1	3	8	0	4
Worker 2	5	7	0	2
Worker 3	0	0	8	0
Worker 4	10	14	0	13

Step 3: Find minimum number of lines that covers all zeros.

	Job 1	Job 2	Job 3	Job 4
Worker 1	3	8	0	4
Worker 2	5	7	0	2
Worker 3	0	0	8	0
Worker 4	10	14	0	13

Step 4: Two lines. Find minimum uncovered (Θ).

	Job 1	Job 2	Job 3	Job 4
Worker 1	3^{-}	8^{-}	0	4^{-}
Worker 2	5^{-}	7^{-}	0	2^{-}
Worker 3	0	0	8^{+}	0
Worker 4	10^{-}	14^{-}	0	13^{-}

	Job 1	Job 2	Job 3	Job 4
Worker 1	**1**	**6**	**0**	**2**
Worker 2	**3**	**5**	**0**	**0**
Worker 3	**0**	**0**	**10**	**0**
Worker 4	**8**	**12**	**0**	**11**

Step 5: Find minimum number of lines that covers all zeros.

	Job 1	Job 2	Job 3	Job 4
Worker 1	**1**	**6**	**0**	**2**
Worker 2	**3**	**5**	**0**	**0**
Worker 3	**0**	**0**	**10**	**0**
Worker 4	**8**	**12**	**0**	**11**

Step 6: Three lines. Find minimum uncovered (Θ).

	Job 1	Job 2	Job 3	Job 4
Worker 1	**1^{-}**	**6^{-}**	**0**	**2**
Worker 2	**3^{-}**	**5^{-}**	**0**	**0**
Worker 3	**0**	**0**	**10^{+}**	**0^{+}**
Worker 4	**8^{-}**	**12^{-}**	**0**	**11**

	Job 1	Job 2	Job 3	Job 4
Worker 1	**0**	**5**	**0**	**2**
Worker 2	**2**	**4**	**0**	**0**
Worker 3	**0**	**0**	**11**	**1**
Worker 4	**7**	**11**	**0**	**11**

Step 7: Find minimum number of lines that covers all zeros.

	Job 1	Job 2	Job 3	Job 4
Worker 1	**0**	**5**	**0**	**2**
Worker 2	**2**	**4**	**0**	**0**
Worker 3	**0**	**0**	**11**	**1**
Worker 4	**7**	**11**	**0**	**11**

Step 8: Four lines – stop the algorithm.

	Job 1	Job 2	Job 3	Job 4
Worker 1	**0**	**5**	**0**	**2**
Worker 2	**2**	**4**	**0**	**0**
Worker 3	**0**	**0**	**11**	**1**
Worker 4	**7**	**11**	**0**	**11**

Using the algorithm described above, it is possible to find an optimal assignment in any two-dimensional matrix. But if the problem cannot be described by a square matrix, there is need to add the missing attributes so it will be possible. The values for these attributes are set so as not to distort the solution. Usually, these are the values for which not worth to do the assignments. This was done so that they are matched last. If we are looking for the maximum cost, then it will be zero. If we are looking for the minimum cost then they are the "infinity".

7.3. Generalized Hungarian algorithm

7.3.1. Problem formulation

The task is to solve the problem of multidimensional assignment. In contrast to problem described above, where there is an assignment only between workers and jobs, we want to add extra dimensions. Such as for example tools. It is possible to solve such problem by reducing it to series of two dimensional problems. For example first resolve assignment problem between workers and job, and next between jobs and tools. But what if we change the order of assignments? For example based on cost matrix how each worker is predisposed to each tool, make assignments between tools and workers, and only then the assignment between workers and jobs. We can get different results, and there is no direct method to determine which one will be better.

We therefore present a problem as the cost matrix, but as a cost cube (Figure 16). Three dimensions represent jobs, workers and tools. Cells of such structure contains combined cost of hiring a worker at particular job using particular tool. For a cube of size N, the result will be a set of N cells (unique in each dimension), which gives the smallest cost. Adding another dimension analogously, we can generalize the problem definition.

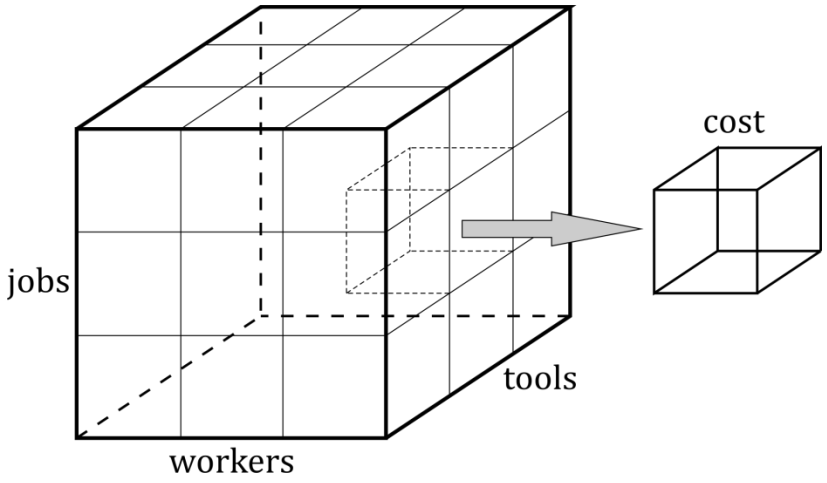


Figure 16. Example of multidimensional assignment problem.

Multi-dimensional assignment problem (MAP) is sometimes referred as multi-index assignment problem can be defined as natural extension of linear assignment problem with minimization of cost function or problem of finding cliques in d-

partite graphs. In very simple words MAP is a higher dimensional version of linear assignment problem, which is defined as follows:

$$\begin{cases} \min \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} c_{ij} x_{ij} \\ s.t. \sum_{i=1}^{n_1} x_{ij} = 1, i = 1,2, \dots, n_1 \\ \sum_{j=1}^{n_j} x_{ij} = 1, j = 1,2, \dots, n_2 \\ x_{ij} \in \{0,1\} \end{cases}$$

Where x_{ij} is a decision variable and is defined as:

$$x_{ij} = \begin{cases} 1 & if\ worker\ j\ is\ assigned\ to\ job\ i \\ 0 & otherwise \end{cases}$$

Multidimensional assignment problem as extension of linear assignment problem is defined as follows:

$$\begin{cases} \min \sum_{i_1=1}^{n_1} \dots \sum_{i_d=1}^{n_d} c_{i_1 \dots i_d} x_{i_1 \dots i_d} \\ s.t. \sum_{i_2=1}^{n_2} \dots \sum_{i_d=1}^{n_d} x_{i_1 \dots i_d} = 1, \quad i_1 = 1,2, \dots, n_1 \\ \sum_{i_1=1}^{n_1} \dots \sum_{i_{k-1}=1}^{n_{k-1}} \sum_{i_{k+1}=1}^{n_{k+1}} \dots \sum_{i_d=1}^{n_d} x_{i_1 \dots i_d} = 1, \\ i_k = 1,2, \dots, n_k, k = 2, \dots, d-1 \\ \sum_{i_1=1}^{n_1} \dots \sum_{i_{d-1}=1}^{n_{d-1}} x_{i_1 \dots i_d} = 1, i_d = 1,2, \dots, n_d \\ x_{i_1 \dots i_d} \in \{0,1\} \end{cases}$$

Where:

- d – is a number of dimension
- n_k- is a number of attributes in dimension k
- $n_1 \leq n_k, k = 2, \dots, d$

In contrast to LAP witch is solvable in polynomial time, MAP is known to be NP-hard problem. This is caused by total number of coefficient:

$$\prod_{k=1}^{d} n_k$$

As well as number of feasible solutions:

$$\prod_{k=2}^{d} \frac{n_k!}{(n_k - n_1)}$$

7.3.2. Related work

Multidimensional assignment problem is first mentioned in literature in 1968 by William Pierskalla [30]. Author define problem using tree where possible solutions are representing by paths in it. Algorithms iterates over all feasible paths and finds an optimal solution. The most interesting thing in the article is that despite the very early years, the algorithm has been implemented and tested on a Univac 1107 computer.

After Pierskalla work there was vast number of application of MAP in literature. In 1994 Poore [31] and four years later Murphey et al. [32] used it for multisensor multitarget tracking. In 1996 Pusztaszeri et al. [33] found it useful in tracking of elementary particles. In 1998 Veenman et al. [34] used it in image recognition. For now there is a lot of algorithms and application of MAP, and its survey [35, 36, 37].

7.3.3. Hungarian algorithm

Hungarian algorithm solves the problem of matching in two-dimensional matrix or bi-partite graph. Such approach allows to assign bi-clusters from the two methods or two different experiments under the same method. However, if there are N results for which we want to fit bi-clusters, the cost matrix is transformed into a hypercube with N dimensions. If we want to find corresponding bi-cluster between two independent experiments we want to maximize the following function:

$$S(R_1, R_2) = \sum_{l=1}^{K} S_{Jacc}(B_l^1, B_{l'}^2)$$

Where R_1 *and* R_2 are two independent bi-clustering experiments and B_l^1 *and* $B_{l'}^2$ are pairs of bi-clusters such that B_l^1 is $l'th$ bi-cluster from result R_1 and $B_{l'}^2$ is bi-cluster corresponding to it from result R_2.

We want to marge N number of bi-clustering results, so there is need find an assignment such that the following function is maximized:

$$S(R_1, \dots, R_N) = \sum_{l=1}^{K} \sum_{i=1}^{N-1} \sum_{j>i}^{N} S_{Jacc}(B_l^i, B_{l'}^j)$$

In other words, to form one of K group, we want to choose from every result one bi-cluster, in such a way that all were so similar as possible within a group. The formula

$$\sum_{i=1}^{N-1} \sum_{j>i}^{N} S_{Jacc}(B_l^i, B_{l'}^j)$$

represent similarity between all pairs of bi-clusters within $l'th$ group. I we have N bi-clustering experiments each of witch with K bi-cluster, the value of function $S(R_1, \dots, R_N)$ is from range:

$$0 \leq S(R_1, \dots, R_N) \leq K * \binom{N}{2} = K * \frac{N!}{2(N-2)}$$

This means that if output is equal 0, than there are N completely different results. And if output is equal to $K * \binom{N}{2}$, then all N results are identical.

7.3.4. Two-dimensional approach

Finding an optimal solution in matching N solution comes down to the analysis of in N-dimensional space. But it can be safely assumed that bi-clustering experiments which are carried out on the same data with the same number of bi-clusters should be similar to each other. Therefore, in order to minimize the computational complexity, the problem can be reduced to a two dimensional space. Rather than representing the cost matrix as a cube in three dimensional space ($\mathbf{R}^3$) or hypercube in general case in n-dimensional space ($\mathbf{R}^n$) more reasonable from complexity points of view will be putting results in a series. In this method, data is presented as N-1 connected bipartite graphs (Figure 17), and N-1 Munkres assignments are performed. Function which it minimizes simplifies a little and looks like this:

$$S_{2D}(R_1, \dots, R_N) = \sum_{l=1}^{K} (S_{Jacc}(B_l^1, B_{l'}^2) + S_{Jacc}(B_{l'}^2, B_{l''}^3) + \cdots + S_{Jacc}(B_{l^{(N-2)}}^{N-1}, B_{l^{(N-1)}}^{N}))$$

Where B_l^1 is $l'th$ bi-cluster from result R_1 and $B_{l'}^2$ is bi-cluster corresponding to it from result R_2. Next $B_{l''}^3$ is a bi-cluster from result R_3 corresponding to bi-cluster $B_{l'}^2$. And so on. Figure 6 illustrates this algorithm. Hungarian algorithm is performed on first pair of results. Then, the third result is added, and Hungarian algorithm is performed between the second and third. The procedure is repeated until all the results will be added.

Function $S_{2D}(R_1, \dots, R_N)$ is from range:

$$0 \leq S_{2D}(R_1, \dots, R_N) \leq K * (N - 1)$$

The upper values the functions S and S_{2D} denote the number of assignments (execution of the Hungarian algorithm) that should be done to assess the quality of the overall fit. Value of $K * (N - 1)$ (bi-clusters are compared only within neighboring results) is usually much smaller than $K * \binom{N}{2}$ (all bi-clusters in the group are compared with each other), and the quality of this approach can be a bit lower than the general approach because it search a local minimum.

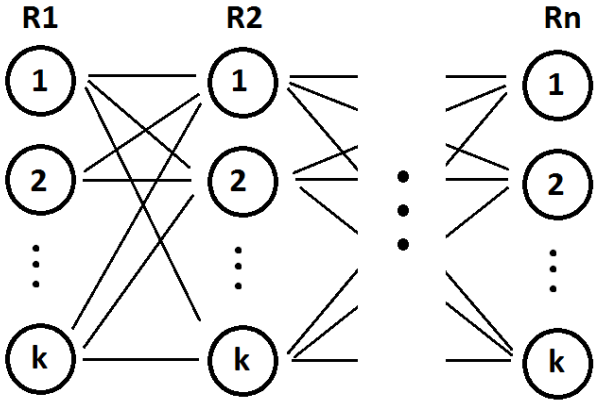


Figure 17. The combination of n independent bi-clustering results with k clusters.

After performing Hungarian algorithm on each pair of neighboring results, K "chains" of bi-clusters are obtained. Each consisting of N bi-clusters derived from the one of N results. This final assignment is influenced mainly by placement of results - the sequence is crucial, but not always. If all the results are very much similar to each other - then the order may not be relevant, and the solution is then optimal.

Example:

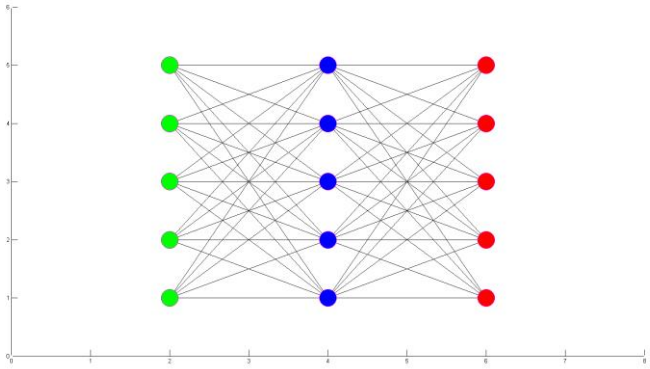


Figure 18. Graphical representation of initial graph with results.

Let's consider three results, each derived from experiments carried out on the same data with the same number of bi-clusters. There are three results: R_1 (green), R_2 (blue) and R_3 (red). First step of algorithm is to form two bi-partite graphs. First graph is made by connecting every bi-cluster B_l^1 from result R_1 with every bi-cluster B_l^2 from result R_2. In the next step add to this second bi-partite graph by connecting every bi-cluster B_l^2 from result R_2 with every bi-cluster B_l^3 from

result R_3 ($l \in \{1,2,3,4,5\}$). The end result is shown in Figure 19. Number of connection (similarities to compute for cost matrices) will amount to 50.

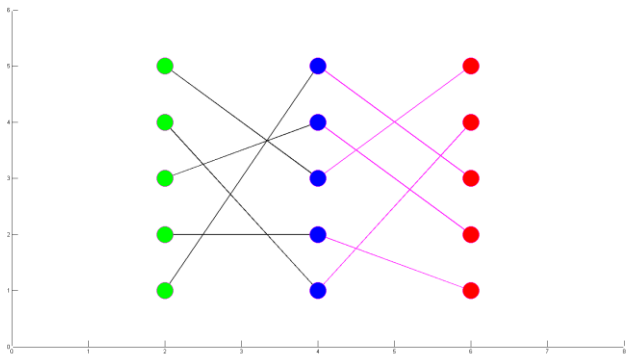


Figure 19. Graphical representation of graph after analysis.

After building bi-partite graphs, third step of this algorithm will be performing Hungarian algorithm two times. First execution will remove unnecessary edges between the results R_1 and R_2, leaving only those that represent best assignments between bi-clusters from this results. Second execution will remove unnecessary edges between the results R_2 and R_3, leaving only those that represent best assignments between bi-clusters from this results.

The remaining edges form the following solution:

$$G_1 = \{B_1^1, B_{1'}^2 = B_3^2, B_{1''}^3 = B_1^3\},$$

$$G_2 = \{B_2^1, B_{2'}^2 = B_5^2, B_{2''}^3 = B_2^3\},$$

$$G_3 = \{B_3^1, B_{3'}^2 = B_2^2, B_{3''}^3 = B_4^3\},$$

$$G_4 = \{B_4^1, B_{4'}^2 = B_1^2, B_{4''}^3 = B_3^3\},$$

$$G_5 = \{B_5^1, B_{5'}^2 = B_4^2, B_{5''}^3 = B_5^3\},$$

Not always, however, the user has such a comfortable situation. Individual results may vary in terms of the number of returned bi-clusters. Such a situation is shown in Figure 20. Each $i'th$ result consists of exactly k_i bi-clusters. Where $i \in \langle 1, N \rangle$.

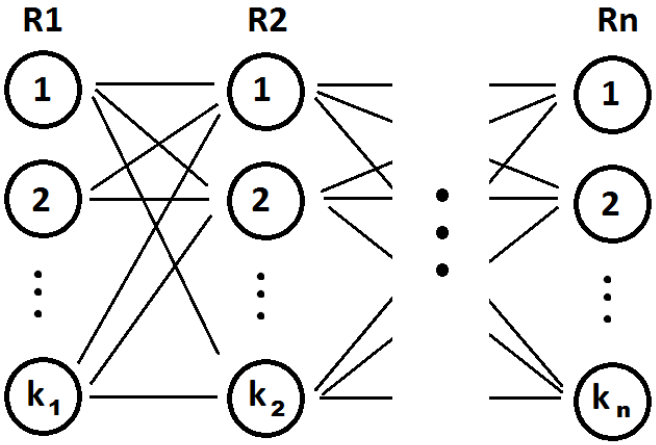


Figure 20. The symbolic diagram showing connected results (with various sizes).

To effectively analyze such data must use the following pre-processing:

1. Sort the results by number of bi-clusters (descending).
2. For each result i, add the following number of empty clusters:

$$k_{max} - k_i$$

Where:

k_{max} – the maximum number of bi-clusters in a single result

k_i – number of bi-clusters in $i'th$ result where $i \in \langle 1, N \rangle$

3. Perform a standard analysis, as described above.

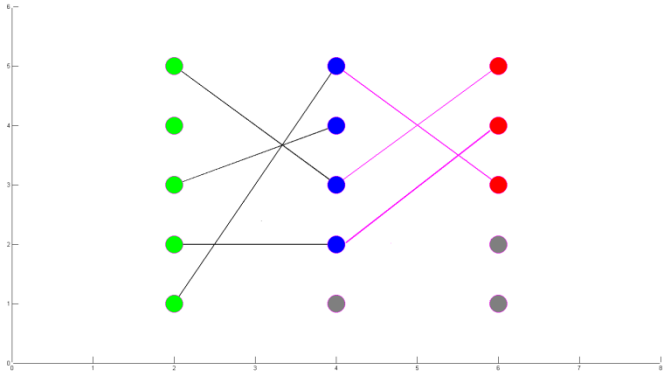


Figure 21. Graphical representation of graph (with empty clusters) after analysis.

Sorting as a way to maximize the number of bi-clusters between "neighboring" results. Additional clusters are empty so that the S_{Jacc} between it and any non-empty bi-cluster was equal 0. This will allow to combine them with others only when it is absolutely necessary due to lack of other options.

Figure 21 shows an example of matching the unbalanced results. First result consist of five bi-clusters, second result consist of four and last result of only three. They are already sorted. Empty clusters are marked in gray and deliberately left unconnected because it would not affect the resulting set anyway.

The biggest drawback of the algorithm described in this subsection is susceptible to changes resulting from the change in order. Poorly matched neighboring experiments (if they are located not at the end) can completely spoils the final assignments. One can protect against this by computing consensus score (chapter 7.4) between each pair of experiments. And then sort experiments by this measure. Because pairs with higher similarity measures should less impair final result.

7.3.5. Multidimensional approach

Two dimensional approach focus on finding some local minimum. But if data are more demanding we have to try little bit harder in terms of time and memory complexity. To find optimal solution we have to find such assignment in witch similarity between all bi-clusters in group is optimal (not only similarity between "neighboring" results like in two-dimensional approach).

Our goal is to find K groups consisting of N bi-clusters, each coming from different result (bi-clustering experiment). All possible combinations of such groups is therefore K^N. We can present a data matrix as a hypercube in N-dimensional space – "cost hypercube". Each element of that hypercube has a value equal to average similarity of bi-cluster over the group that it represent:

$$\overline{S_{Jacc}}_a = \frac{\sum_{i=1}^{N} \sum_{j=i+1, j\neq i}^{N-1} S_{Jacc}(B_a^i, B_a^j)}{\binom{K}{2}}$$

Where:

- a is single element from "cost hypercube" consisting of N bi-clusters B_a^i, where $i = 1, \dots N$,
- B_a^i – is bi-cluster coming from result $i'th$, and being part of element a
- $\overline{S_{Jacc}}_a$ is average Jacard similarity in group a

Assignment should be unique in this respect that no bi-cluster can participate in more than one resulting group. The solution will therefore consist of the K elements, and number of all possible solutions will be $N!^K$. It is far beyond naive method.

This multidimensional approach of Hungarian algorithm is based on translating König's theorem from two dimensional space to n-dimensional space. And on that basis the pseudo-code of translated Hungarian algorithm is as follows:

- `Step 1 and 2 become Step 1, 2, 3, …, N`
  `In every Step i (where i = 1,…,N) From cost martix in hyperplane formed after deduction of dimension „i" we subtract its minimum value`

- Step N+1: Choose the minimum number of hyperplanes to cover all zeroes. If this number = N, STOP — an assignment can be made.
- Step N+2: Determine the minimum uncovered number (of numbers that do not lie on any hyperplane.) (call it Θ).
  - Subtract Θ from uncovered numbers.
  - Add d to numbers covered by two lines.
  - Numbers covered by one line remain the same.
  - Then, GO TO STEP N+1.

And pseudo code for resolving problem in Step N+1:

- Finding the Minimum Number of Lines and Determining the Optimal Solution
  - Step 1: Find a dimension with only one unlined zero and circle it. (If all dimensions have two or more unlined zeroes choose an arbitrary zero.)
  - Step 2: If the circle is in a row with one zero, draw a line through its column. If the circle is in a column with one zero, draw a line through its row. One approach, when all rows and columns have two or more zeroes, is to draw a line through one with the most zeroes, breaking ties arbitrarily.
  - Step 3: Repeat step 2 until all circles are lined. If this minimum number of lines equals m, the circles provide the optimal assignment.

Example:

Let's consider three results, each derived from experiments carried out on the same data with the same number of bi-clusters. There are three results: R_1 (green), R_2 (blue) and R_3 (red). The first step of the algorithm is connect all the bi-clusters between methods - each with each. In Figure 9 is an example of the three methods, and 15 bi-clusters as a whole, so the number of connections in this case will amount to 75. What we really look at is the triangles formed by the vertices

coming from different results. All combinations of such triangles is 125, and they form a cube with dimensions 5x5x5.

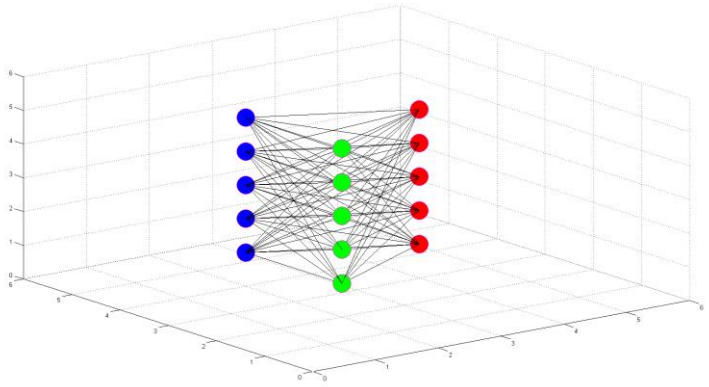


Figure 22. Visualization of original data before analysis.

After building cost hypercube, next step of this algorithm will be performing Hungarian algorithm on it. The result will be 5 groups, each consisting of 3 bi-clusters. In Figure 10 the solution appears as 5 independent triangles.

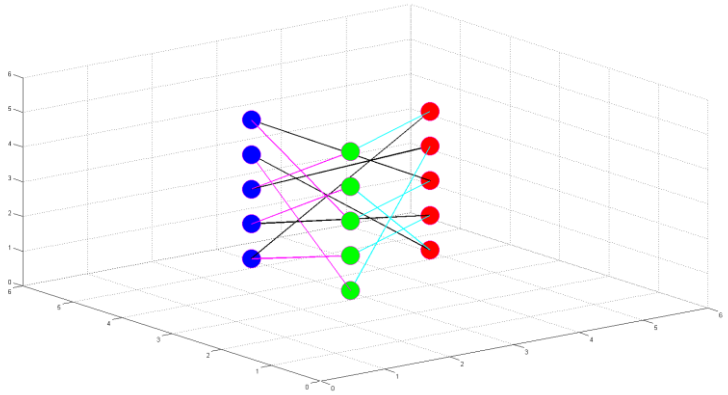


Figure 23. Visualization of original data after analysis.

7.4. Consensus algorithm

A single experiment may give unsatisfactory results due to the fact that chosen method is designed for different bi-cluster structure, or strongly depend on initial conditions and provide not optimal solutions. To find the best solution for the input data, there is need either fully understand the structure of data, or perform many different experiments using as many methods as possible to choose an appropriate. Even the best method, suitable for data structure, in addition to the relevant data may contain noise, or contain incomplete data.

In contrast to relying on single experiments or single methods, this thesis propose a solution focuses on integrating the results into one general and more reliable solution. Each result will contain the correct data (such that should be part of the bi-cluster), and some data which were in it because of the noise in the data, due to the local minimum or other errors. Algorithms assumes that the experiment is performed repeatedly (using different initial conditions and/or different methods), and then the results are combined, with should filter out unwanted data. The final result should consist of K bi-clusters. K may be a number specified by the user or obtained as a result of the calculations. If this number is known, results with less bi-clusters are complemented by empty one, and the results with more bi-clusters are reduced by removing bi-clusters with the lowest quality. Finally each i'th results is looks as follows:

$$R_i = \left\{B_1^i, B_2^i, \dots, B_K^i\right\}, \text{where } B_l^i = (I_l^i, J_l^i), \text{and } l \in 1, \dots, K$$

Where R_i means i'th result where $i \in 1, \dots, N$

Following the experiments, bi-cluster should be grouped to K groups, such that none of the bi-cluster within the group does not come from the same experiment:

$$G_l = \{B_{l'}^1, B_{l'}^2, \dots, B_{l'}^N\}, \text{where } B_{l'}^i = (I_{l'}^i, J_{l'}^i), \text{and } i \in 1, \dots, N$$

Where G_l means $l'th$ group where $l \in 1, \dots, K$. Bi-clusters $\{B_{l'}^1, B_{l'}^2, \dots, B_{l'}^N\}$, are chosen to maximize the following function:

$$\sum_{l=1}^{K}\sum_{i=1}^{N}\sum_{j=i+1,j\neq i}^{N-1} S_{Jacc}(B_l^i, B_l^j)$$

Following the grouping, within each group G_l we merge its bi-clusters to one bi-cluster

$$B^l = (I^l, J^l)$$

In such a way that the vectors I^l *and* J^l were formed from the attributes included in as many bi-clusters from group l as possible. In the most restrictive form in all:

$$I^l = \{x_l \in X : x_l \in I_l^1, x_l \in I_l^2, \dots, x_l \in I_l^N\}$$

$$J^l = \{y_l \in Y : y_l \in J_l^1, y_l \in J_l^2, \dots, y_l \in J_l^N\}$$

This condition can be relaxed by allowing the absence of an attribute in a given number of bi-clusters (This may be a threshold, set as a parameter of the algorithm).

Proposed method assumes a solution in which that threshold is adjusted during the algorithm, to meet parameter MinC (minimum number of attributes in bi-cluster) or MinQ (minimum quality of resulting bi-cluster). This parameter may be a number specified by the user or obtained as a result of the calculations

To summarize the whole process: we have a set of N results, where each is the result of an experiment conducted on the same data matrix with the same number of bi-clusters (k). Algorithm is as follows:

- Using a generalized Hungarian algorithm assign bi-clusters from all methods so as to form K sets, each consisting of N bi-clusters,
- Compute for each bi-cluster one of quality index described in Chapter 5.2.
- In each $k'th$ set, remove bi-clusters with quality index below certain threshold T_1 (parameter set by the user or computed automatically).
- For each $k'th$ set compute average quality index, and remove whole set if its value is below certain threshold T_2 (optional parameter set by the user or computed automatically).
- For each $k'th$ set compute average $n_{i,k}$ (number for $i'th$ attribute, denotes the number of bi-clusters in set k, in which attribute is present), and remove

whole set if its value is below certain threshold T_3 (optional parameter set by the user or computed automatically).

- Match the weight to each attribute i from bi-cluster j taken from set k, such that:

$$W_{i,k} = \frac{n_{i,k} + \frac{Q_{i,k} - \min\limits_{k} Q_k}{\max\limits_{k} Q_k - \min\limits_{k} Q_k} * N}{2}$$

  Where:

  - $n_{i,k}$ – number for each $i'th$ attribute, denotes the number of bi-clusters in set k, in which attribute is present.
  - $Q_{i,k}$ – average value of quality index of bi-clusters in $k'th$ set, which contains attribute $i'th$.
  - $\min_k Q_k$ – minimum value of quality index in $k'th$ set.
  - $\max_k Q_k$ – maximum value of quality index in $k'th$ set
  - N – number of results/elements in sets.

- Set P = N,
- For every set representing single bi-cluster:
  1. Select only those attributes, for witch value of $W_{i,k}$ is equal or greater than P.
  2. If number of attributes in bi-cluster are equal or greater than MinC and/or quality of bi-cluster is equal or greater than MinQ, than stop, otherwise go to 3.
  3. Decrease P, and go to step 1.

8. Graphical presentation of results

8.1. Presenting bi-clusters

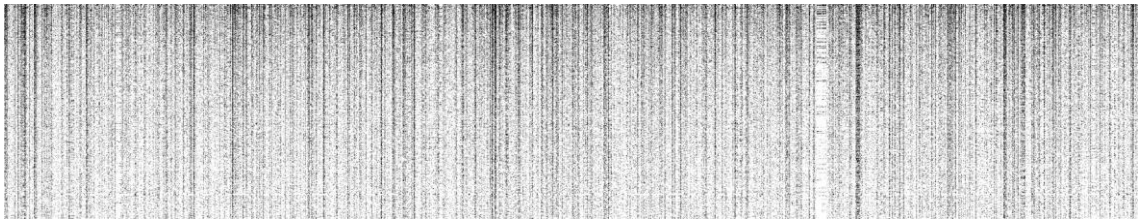


Figure 24. Real data from Monica Chagoyen paper [38].

Real data, regardless of origin (micro-array experiments, document-term frequencies, general text mining data, etc.), at first glance may appear to be random and devoid of any structure. Figure 11 shows a visualization of the data matrix containing the relationship between words and genes. The vertical dimension represent genes, and horizontal dimension the words. At the intersection of these two dimensions is a value denoting number of occurrences of a word in the context of a given gene. Brighter values mean fewer occurrences, while the darker more. The data has been very carefully chosen to contain eight bi-clusters.

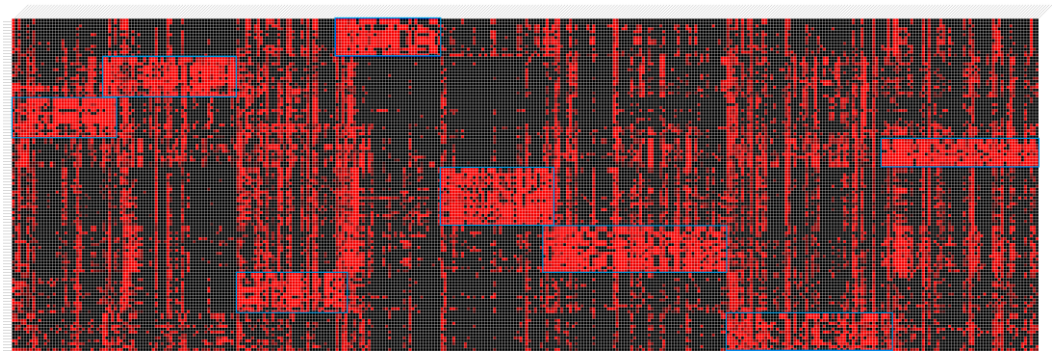


Figure 25. BiVoC algorithm sample result.

To reveal hidden structure it is necessary to reorder rows and columns. The literature contains many examples of algorithms for implementing this task.

8.1.1. BiVoC

BiVoC stands for Bi-dimensional Visualization of Clustering and it is a part of package Biorithm [39]. It is a set of tools written in C++ designed to analyze data mainly in molecular systems biology. This software is developed by T.M. Murali's

research group and is created for a several years. BiVoC is a part of this work, and it is an algorithm for laying out bi-clusters in two-dimensional matrix. It takes on input data matrix and information about computed bi-clusters. As the very first step algorithm removes from data matrix all irrelevant rows and columns (those not involved to any bi-cluster). After filtering attributes method perform reordering to group rows and columns, so that those who are involved in the same bi-cluster appeared next to each other. Example result is shown on Figure 25.

8.1.2. BicOverlapper

BicOverlapper is visualization tool introduced by Rodrigo Santamaria, et al. [40] in 2008. They proposed approach based on undirected graph, where bi-clusters are plotted as complete sub-graphs (Figure 26). Edges consist of rows and columns from original data matrix.

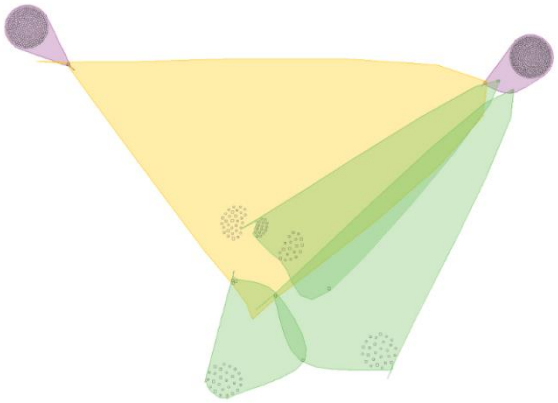


Figure 26. BicOverlapper graph representation.

For clarity, the edges of the graph are not drawn. Nodes belonging to bi-clusters are gathered into rounded shapes. Each pair of nodes from the . The main advantage of this tool is that visualization is not static. User can interact with it, and change parameters of a model, BicOverlapepr layout, etc.

8.1.3. BiCluster Viewer

In 2011 Julian Heinrich et al. [41] proposed tool for visualizing bi-clustering results from gene expression data analysis. Authors draw bi-clusters using heat maps representation, and what is very interesting, allow for duplicate columns and rows. Heat maps data values mapped to grayscale values using linear interpolation

between smallest and largest value of the original data matrix. The algorithm allows the duplication of rows and columns to make sure that all of them are located in contiguous regions.

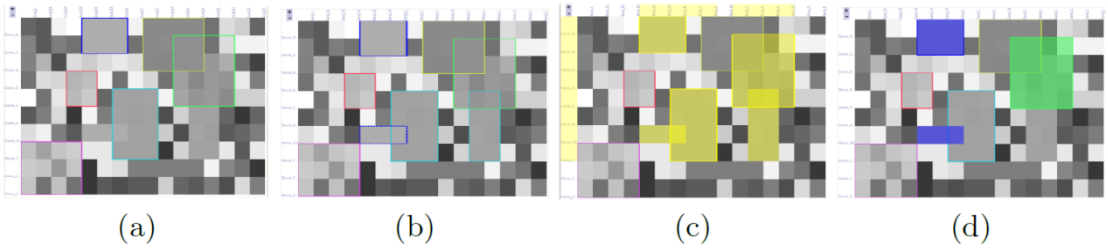


Figure 27. Example of BiCluster Viewer, taken from original publication [41].

Figure 27 shows en example of presenting toy example in four different representations. First view (a) is default and represent each bi-cluster by its major rectangle only. In second mode (b) all bi-clusters are represented. In third (c) view is with tree highlighted bi-clusters, and last (d) with permanently highlighted bi-clusters

8.2. Presenting the results of domain

Sometimes determining the bi-clusters is not enough, the need is to determine their quality also. For this purpose it is necessary to interpret the obtained clusters, and to determine their quality according to the field to which they belong.

8.2.1. Clusters containing genes

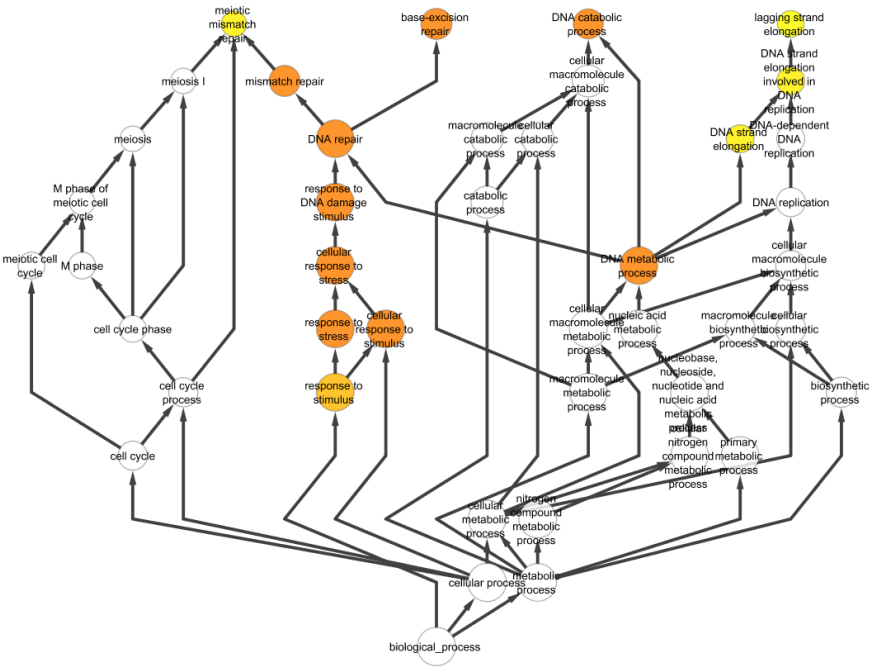


Figure 28. Gene ontology tree composed with gene ontology terms.

To asses quality of gene cluster we use gene ontology database. Genes clusters are connected with gene ontology terms. Next step is to using those term build network (Figure 28). For this purpose Cytoscape program [42] with Bingo plugin [43] is used. Assumption is that genes strongly correlated with each other, will lead to small and dense trees, because they shouldn't be associated with very diversified group of terms.

In Figure 28, only colored terms are the result of the analysis. White one are used only for visualization purposes, to connect resulted terms with root.

8.3. Presenting the results from different experiments.

It's very useful to compare result coming from different experiments performed within the same or different method. This is especially useful when there is need to examine how repeatable methods are or merge different results.

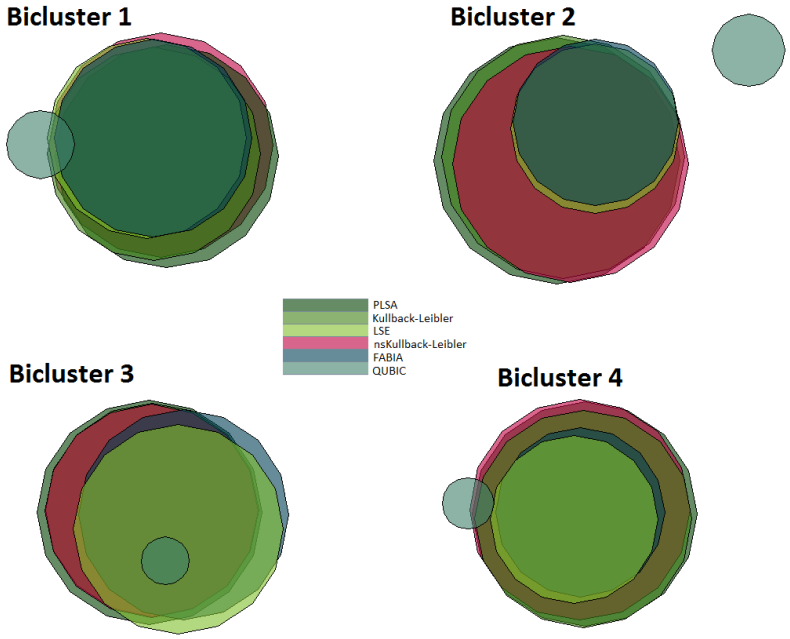


Figure 29. Venn Diagram with visualization of merge of different results. Computed using VennMaster tool [44].

To begin any analysis first should single bi-clusters between the methods be associated. Wide description of how this can be done is presented in chapter 7.4. As a result there are sets of bi-clusters, which should have a high level of similarity. Mentioned in this paragraph analysis is intended to visualize this similarity, so that it was easily evaluable by the user.

Similarity between set can be easily visualized by plotting paired sets on Venn diagrams. Example of such visualization is presented on Figure 29. There are four different bi-clusters set from six different experiments. From such analysis be done conclude similarity level with respect to size.

9. Computational experiments

9.1. Environment for data generation and evaluation

For the purpose of this PhD thesis was created software named AspectAnalyzer. Its distributed system written in C# programming language and .NET Framework. It has implemented several algorithms taken from literature and consensus methods described in this thesis. Graphical user interface is based on Windows Presentation Foundation. Communication within program and within different instances of AspectAnalyzer on different nodes is based on Microsoft MSMQ queues and all mathematical computation are done using ILNumerics.

Figure 30. AspectAnalyzer main window.

ILNumerics is a high performance math library available on GPL Public license. Library extends .NET Framework with tools for scientific computing, provides simplified and optimize code for matrix operations. Below table (Table 3) shows differences between standard C# implementation and analogous implementation using ILNumerics. Tests were done using AspectAnalyzer program and presented times are for execution of one pass of the loop. In some cases dedicated library was twenty times faster than regular implementation.

Table 3. Comparison of standard C# implementation and ILNumerics.

Method	**C# [s]**	**ILNumerisc [s]**
PLSA	14	2
Kullback-Liebler	12	0.7
Least Square Error	7	0.5
NonSmooth Kullback-Liebler	24	0.9

9.1.1. Data

Due to the distributed nature of the system, data is stored in Microsof SQL database. Figure 31 shows diagram of AspectAnalyzer database, and data is divided into two groups – data related to matrices and data related to results. In first group we can find matrices with its data and all description such as matrix noise level, bicluster numbers, etc. All matrices comes also with type which can be set to V matrix (orginal data matrix) and optionally (is algorithm perform matrix factorization) W matrix (left matrix from factorization) and H matrix (right matrix from factorization). There is no limit on number of different properties. To add one there is only need to add its description to PropertiesTypes table.

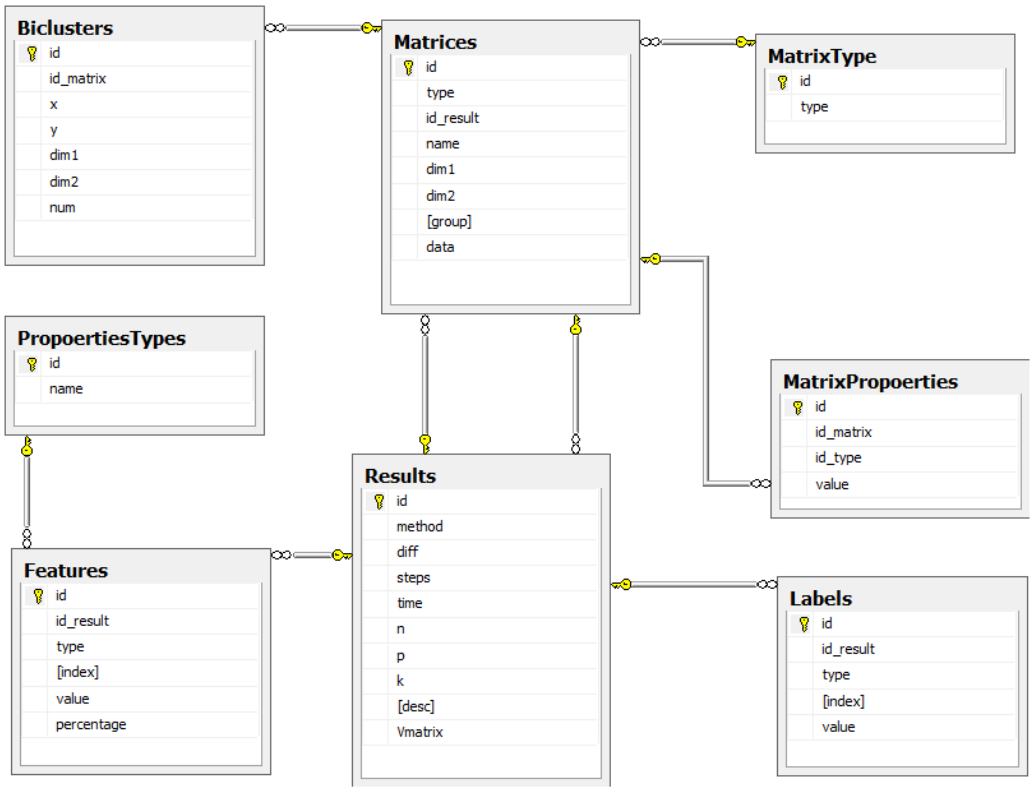


Figure 31. AspectAnalyzer data diagram.

Schema related to results contains a little bit more information's. Table "Results" contain detailed data about single experiment (such as number of steps, value of distance function, etc.). With results there is also a room for features computed after experiments using for example factorized matrices. Such can be estimated bi-

clusters, values of quality indexes, etc. Data was tested on version 2008 R2 and 2012 of SQL Server in Express Edition. In such configuration program and all its features is free and available for all operating system using .NET Framework and MSMQ queues. However in free version of SQL server Express only limitation for user, that is important from AspectAnalyzer point of view, is that database size is limited to 10GB. For comparison, commercial versions have the limit set to 524PB. But free version is sufficient for many application of AspectAnalyzer, and if not there is a possibility to divide large database into smaller parts to omit restrictions.

9.1.2. Distributed computing

Thanks to the use of the database not integrated with the program, there is an opportunity to build a distributed system. It is possible to run many instances of AspectAnalyzer on a different nodes, different location etc. All instances can be set to master-slave model in which one instance is master node, and all others should be in slave mode. All nodes report to master every 5 seconds with information about current load, completeness of current tasks etc. Master node can manage remotely by sending specific instructions to slave-node using its IP address.

Figure 32. Node Manager window from AspectAnalyzer.

Using Node Manager panel shown on Figure 32 user can specify tasks, define experiments and system will automatically balance those jobs over running instances taking into account current load, number of cores, etc. Remote steering has the same abilities as normal one, and whole communication is done using MSMQ, so only one limitation is that ports on nodes IP should by open between every slave node and master node.

Besides defining tasks and balancing it on all connected nodes, Node manager can also change all possible settings of every connected slave (Figure 33).

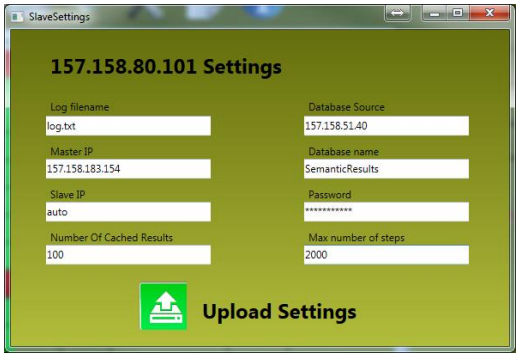


Figure 33. Slave settings window for node manager.

9.1.3. Defining own synthetic matrix

User can define its own synthetic matrix with its all relevant properties. Procedure starts with defining size of data. After this, on a screen appears black border which limits data matrix. Within those borders it is possible to place bi-clusters as colored squares. For each there is an option which allows to defined data structure inside bi-cluster. User can choose one of predefined settings or load sample bi-cluster from text file.

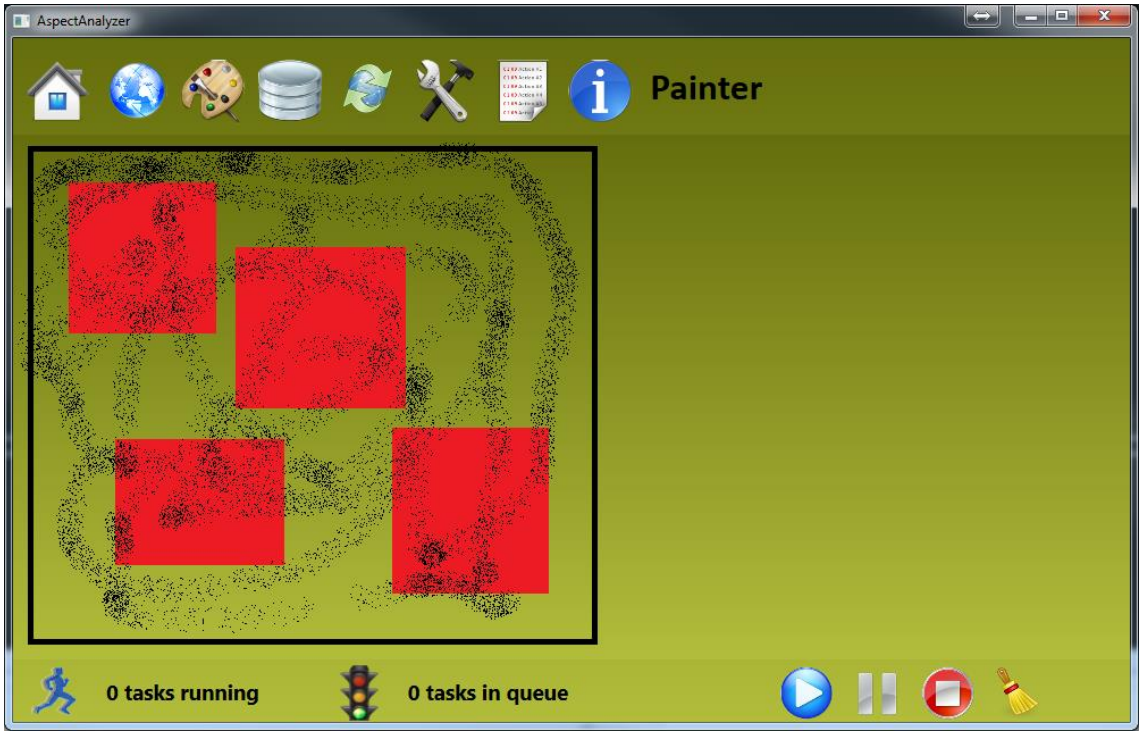


Figure 34. Painter window from AspectAnalyzer.

After defining bi-cluster position, structure and size there is also possibility to introduce noise to data matrix. Noise level can be generated inside and/or outside of bi-cluster automatically, by defining its level and characteristic. And there is also possibility to generate noise manually by printing it using special "spray" control. Level is generated manually in this mode, but user is able to define noise characteristics such as average value, distribution, etc.

9.1.4. Browsing data and results

Using form presented on Figure 35 user is able to browse over results stored in database. Main window shows only general view with list of data matrices and summary number of results for it. Double click on matrix results with loading it to main screen and options with defining bi-clustering experiments. Other way is to clicking "chart and notes" icon which for the selected matrix displays in the table a more detailed summary. It contains results grouped by method and number of bi-clusters with average, minimum and maximum value of divergence function (if such function exist for selected method). The third level of nesting, available under an icon mentioning above, is a view of the individual results.

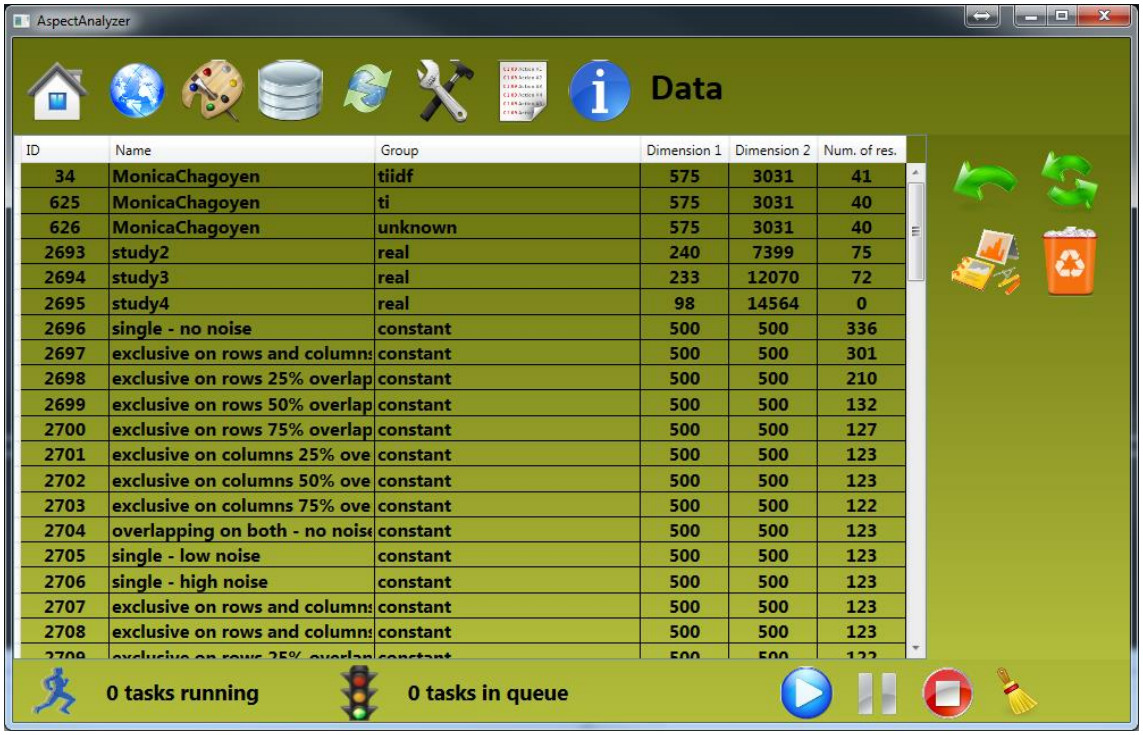


Figure 35. Data window from AspectAnalyzer.

On detailed window with list of single results last available option is calling window with result description. On such window user can draw chart with divergence function (Figure 36), extract customized bi-clusters from result or compute quality indexes for it.

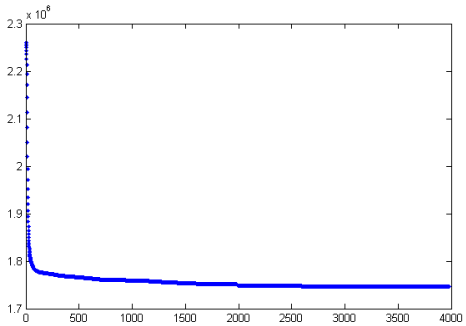


Figure 36. Sample chart with changes in divergence function values.

9.1.5. Update functionality

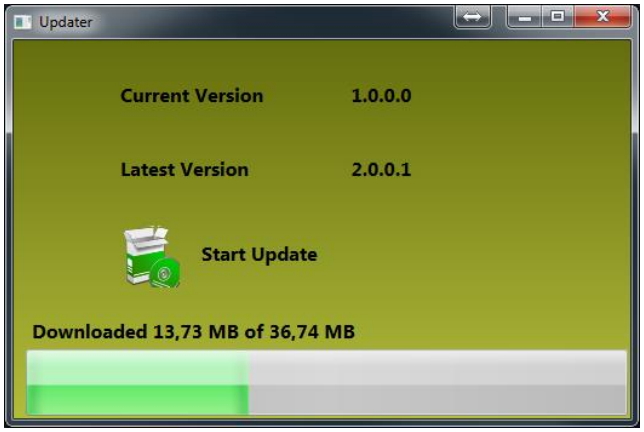


Figure 37. Aspect Analyzer Update Window.

Since program is available public, and hosted on dedicated website, there is possibility of fully automated update process. Feature is available only if currently installed version is lower that this posted on project site.

9.1.6. Program availability

Figure 38. About window from AspectAnalyzer.

Whole system is based on free and widely available components as ready to use installer posted public on dedicated website http://AspectAnalyzer.foszner.pl/ (Figure 39). Project site in addition to the installation version of the program itself

also contains a comprehensive description and user manuals. Is organized in the form of a blog on which are published up to date information about changes and new versions.

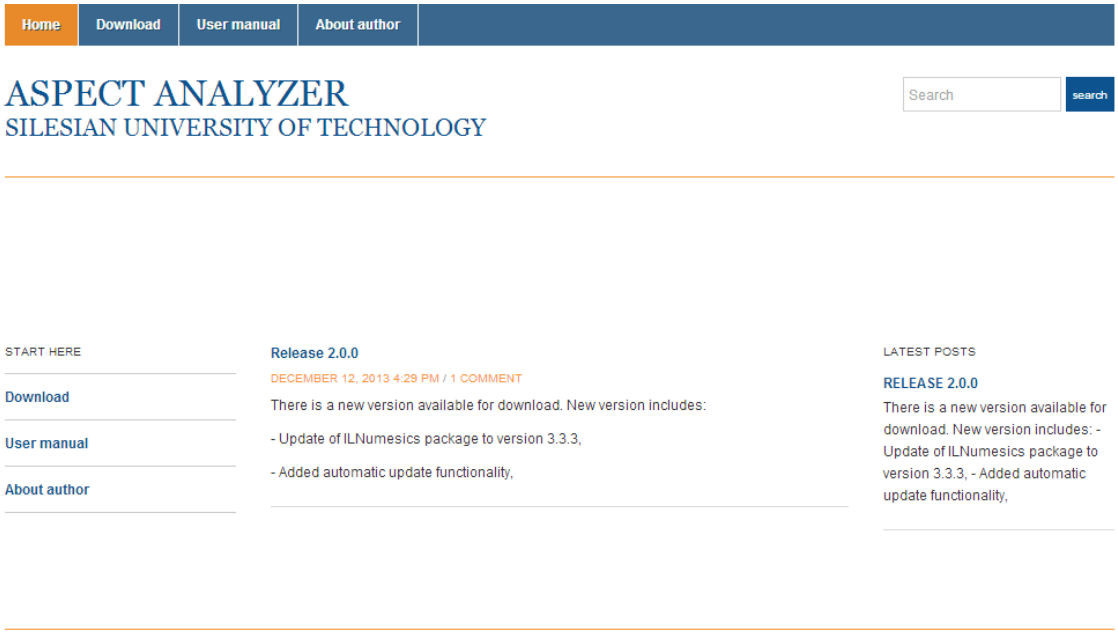


Figure 39. Aspect Analyzer official website.

9.2. Third-party software

During work on this thesis several third-party software were used.

- **Bi-Bench** [18] – Bi-clustering package consist of 3 important layers:
    - User API – written in python set of functions for running all features of package
    - R-CRAN package – for functionalities related with biological data
    - Bi-clustering algorithms package installed separately in the operating system
- **COLASCE** [45]
- **CPB** [19]
- **BBC** [46]
- **QUBIC** [5]
- **VennMaster** [44]
- **BicOverlapper** [40]
- **Cytoscape** [42]
- **Bingo** [43]
- **ILnumerics** [47]
- **BiCluster Viewer** [41]

9.3. Data

9.3.1. Synthetic data

There is a large number of data structures, and very often algorithms from literature specialized only in specific one. In more details all possible structures are described in chapter 5.1 and small survey of bi-clustering algorithms in chapter 6. The main characteristic of synthetic data is its diversity. They contain every important combinations of bi-cluster structures, and the degree to which overlap the rows and columns. Data consist of matrices with one of six major structure each. Additionally every matrix represents single structure appears in one of nine variants regarding to bi-clusters overlapping over rows and columns. That gives 56 matrices in . Figure 40 shows examples from that set.

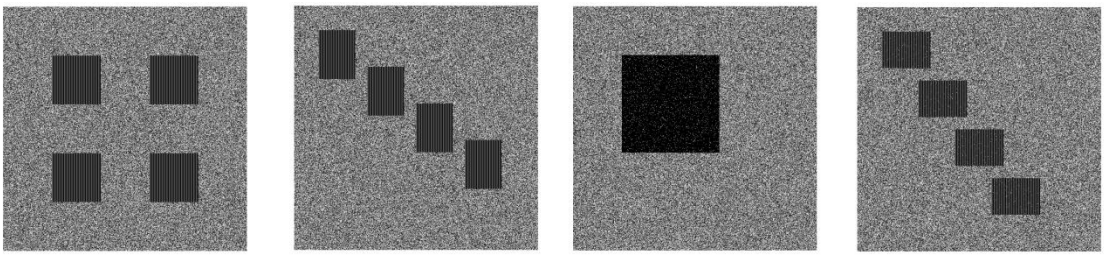


Figure 40. Samples of synthetic data.

Regarding to level of overlapping test set consist of various number for matrices with different variants of bi-clusters positions in data matrix. We distinguish data matrices with:

- Single bi-cluster,
- Bi-clusters with exclusive rows and columns,
- Bi-clusters exclusive on rows and overlapping on columns (25%),
- Bi-clusters exclusive on rows and overlapping on columns (50%),
- Bi-clusters exclusive on rows and overlapping on columns (75%),
- Bi-clusters exclusive on columns and overlapping on rows (25%),
- Bi-clusters exclusive on columns and overlapping on rows (50%),
- Bi-clusters exclusive on columns and overlapping on rows (75%),
- Bi-clusters overlapping on both dimensions (up to 100% of overlap)

Each structure described above appears in six different variants of bi-clusters values. Regarding to bi-clusters structure we distinguish data with (every single matrix contains only one of the following):

- Constant data
- Constant data up-regulated
- Plaid data
- Shifted data
- Scaled data
- Shift and scale data

To sum this up we have nine different data sets regarding to bi-cluster position and six regarding to bi-cluster structure. The final set of the consisting of 56 matrices, each having a different structure and distribution of the bi-clusters.

9.3.2. Real data

9.3.2.1. Text mining data

Real data come from article by Monica Chagoyen, et al [38]. Data were restored based on the informations and the sources from the article. It was a matrix containing the number of occurrences of words in the context of genes. Genes were selected from SGD8 database (*Saccharomyces cerevisiae* genome) and each associated with one of eight broad biological processes (each of which described by GO Ontology term):

- cell cycle (GO:0007049),
- cell wall organization and biogenesis (GO:0007047),
- DNA metabolism (GO:0006259),
- lipid metabolism (GO:0006629),
- protein biosynthesis (GO:0042158),
- response to stress (GO:0006950),
- signal transduction (GO:0007165),
- transport (GO:0006810).

All genes were annotated by the experts with 7080 articles. At least one article with one gene. We download all documents listed in article from PubMed da-

tabase. Single document is constructed by concatenating the titles and the abstracts. After removing very frequent terms (appears in more than 80% of genes), and very rare terms (less than 4%), we obtain 3031 words. Term frequencies were weighted by IDF measure [48], which stands for *inverse document frequency*:

$$IDF_j = \log\left(\frac{T}{t_j}\right)$$

Where

IDF_j – is inverse document frequency for term j,

T – total number of documents in set

t_j – number of documents that contains document j

Than final value of balanced term frequency of term j in document I, can be defined as:

$$D_{ij} = tf_{ij} * IDF_j$$

9.3.2.2. Microarray data

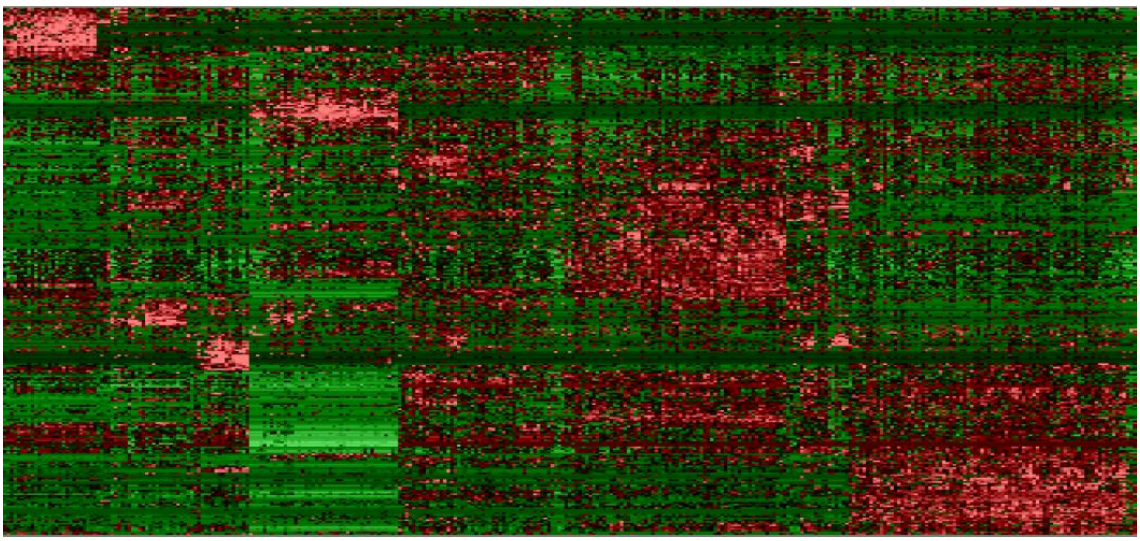


Figure 41. Gene expression data from Eng-Juh Yeoh, at el. [49] presented as heatmap.

Data were taken from Eng-Juh Yeoh, at el. [49] publication is gene expression matrix consist of 360 microarray experiments. Each experiment is taken from different leukemia patient. Each patient in test group has one of six subtype of leukemia. Such define data set could have been consist of six bi-clusters, each associated with a different kind of disease.

9.4. Computational results

9.4.1. Synthetic data

For each matrix described in chapter 9.3.1 were performed 100 experiment – each time with different initial conditions for non-deterministic algorithms and one experiment for deterministic algorithms.

- BBC
- Cheng-Church
- BiMax
- CPB
- FABIA
- XMotifs
- Plaid
- ISA
- Qubic

Each algorithm was run on all the data a hundred times. This gives a total number of few thousands of experiments, shown below, by the kind of data.

For each data, after all analysis, meta-algorithm was performed using results of above nine algorithms. This is denoted on charts by . As clearly seen in the following figures, the algorithm proposed in Chapter 7.4, proved to be the best in most cases on synthetic data.

Results are presented in Appendix A Synthetic data. Outcome from each data matrix are presented graphically in the chart and in the form of a numerical table. The graph in the vertical axis has “Relevance” while on the horizontal “Recovery”. It is desirable that the result which is considered the best was in the upper right corner (1,1), and the worst at the bottom left (0,0).

9.4.2. Real data

For the purpose of this analysis two different data sets were chosen. First set resulted from text mining analysis of publication dataset. Data matrix was the matrix of occurrences of words in the context of genes. Documents describe one of the eight subjects. It is therefore expected eight bi-clusters composed of a set of genes and with a set of words. Second data set was gene-expression matrix coming from microarray experiments performed on patients with leukemia. Each of the patients belonged to one of the six independent groups. The analysis should result with six bi-clusters composed of groups of genes and groups of conditions. Both data sets are described in Chapter 9.3.2.

At the beginning of the experiments are performed by known algorithms. Deterministic algorithms, the result of which is repeatable and does not depend on the initial conditions, are performed once. Analyses using non-deterministic algorithms, or such that the result depends on the initial conditions are repeated a hundred times.

9.4.2.1. Results repeatability

At the beginning the repeatability of results over the experiment are examined. To visualize this bi-clusters from different experiments were assigned to itself by the method described in section 5.4. For this second group additional analysis has been performed in order to filter out algorithms that are not suitable. For such we consider algorithms that return a bi-clusters of a quality below a certain threshold and its results are not repeatable. To visualize this bi-clusters from different experiments were assigned to itself by the method described in section 7.4. Then, using software VennMaster [44] for each bi-cluster there is a diagram that shows how different clusters are between different experiments. In brackets, after the name its name for each bi-cluster is presented the average AVC index value (described in section 5.2.2). The average sizes of a single bi-cluster is represented by two numbers in angle brackets under the diagrams. They represent respectively the cardinality of the first and second cluster (components of the resulting bi-cluster).

For our example visualizing such analysis were selected four non-deterministic algorithms based on the nonnegative matrix factorization. Algorithms

were described in chapter 6.1 and they consist in factorizing the data matrix as the product of two other matrices (denoted by matrix W and matrix H):

$$A \approx W * H$$

Bi-clusters are read directly from those matrices. Where chosen PLSA by Hoffman [50] and three NMFs based on three different distance functions: Least Square Error [8], Kullback-Liebler [8], and nonSmooth Kullback-Liebler [22]. Those algorithms are strongly depend on initial conditions which are randomly generated matrices W and H. For this reason, each experiment was repeated ten times, each time with a randomly filled matrices W and H. Below examples from that analysis:

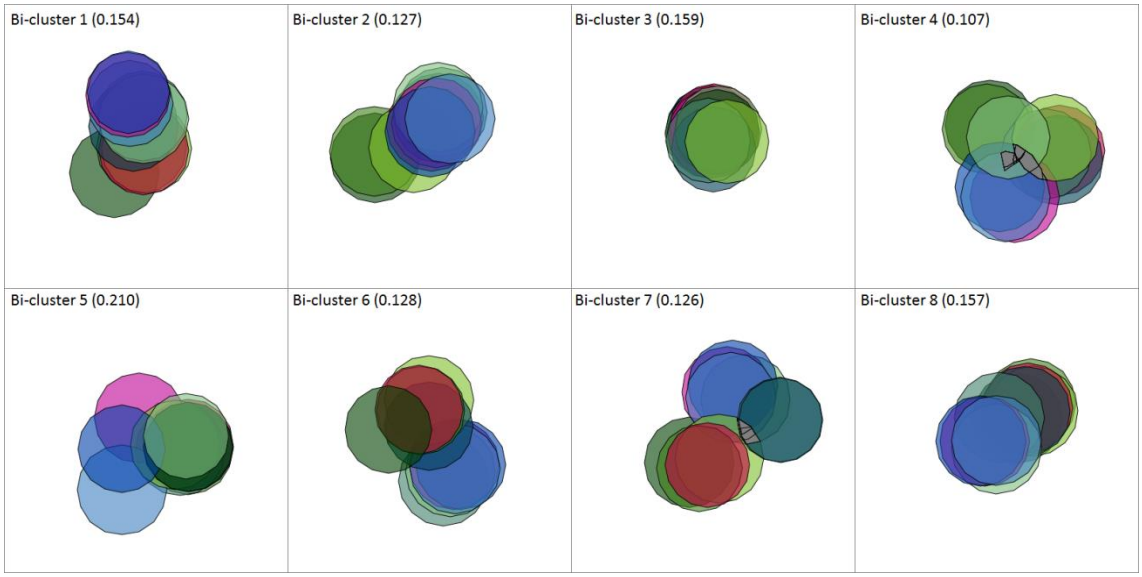


Figure 42. Probabilistic Latent Semantic Analysis.

Average size of single bi-cluster in method on Figure 42 is <34, 38>, and average value for AVC index is 0,138. As there is shown results are moderately good. Only bi-cluster 3 is fully repeatable, and other bi-clusters have different results close enough but not always at the same spot.

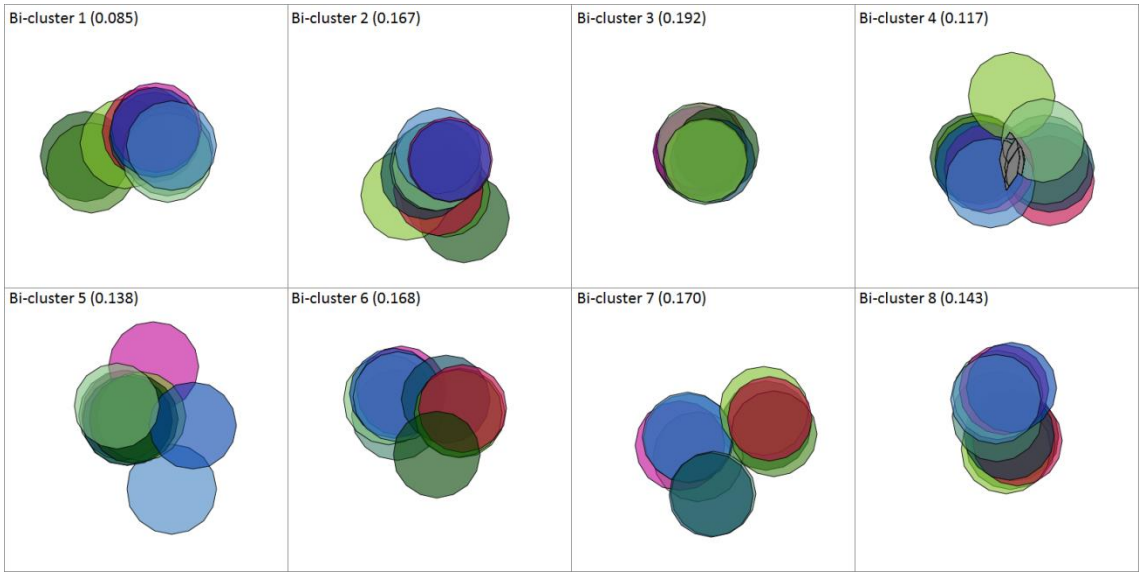


Figure 43. NMF based on Kullback-Liebler divergence function.

Average size of single bi-cluster in method on Figure 43 is <25, 20>, and average value for AVC index is 0,129. Conclusions are very similar to those on previews figure.

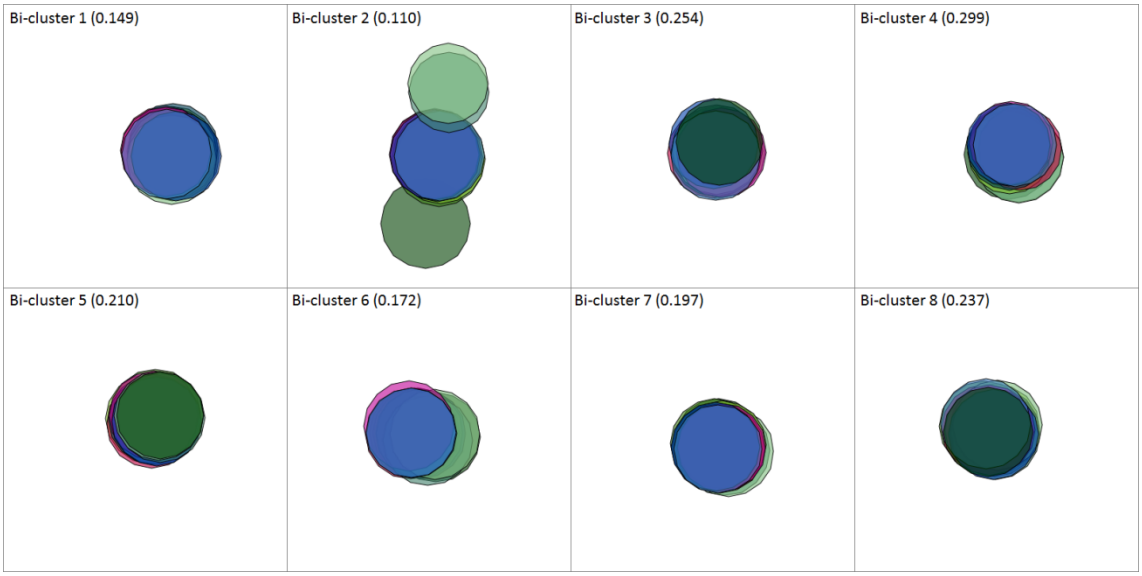


Figure 44. NMF based on Least Square Error distance function.

Average size of single bi-cluster in method on Figure 44 is <69, 75>, and average value for AVC index is 0,211. Above example can be consider as very good, because all bi-cluster at almost every result looks the same. Only exception is at bi-cluster 2 where three results are different.

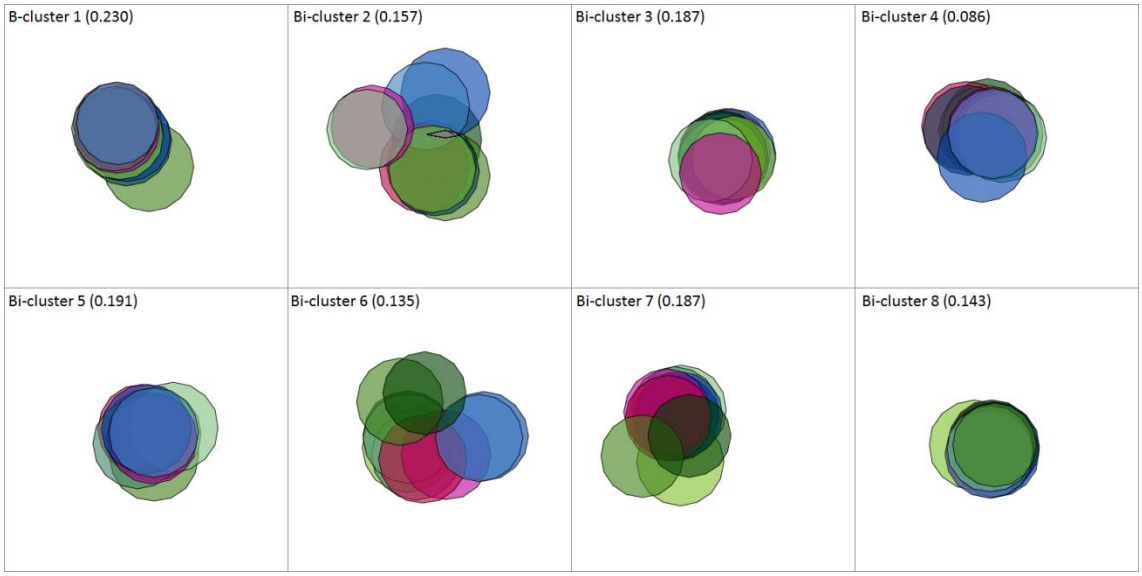


Figure 45. NMF based on non-smooth Kullback-Liebler divergence function.

Average size of single bi-cluster in method on Figure 45 is <46, 39>, and average value for AVC index is 0,253. In this example only half of bi-cluster produce repeatable clusters over results.

9.4.2.2. Consensus algorithm comparison

Table 4 and Table 5 presents differences between normal algorithms from literature and approach based on merging results. For each method bi-clustering experiment were carried out one hundred times for nondeterministic methods, and one for deterministic method . Then using method described in chapter 5.3 For each method separately connection between bi-clusters have been made. This results with eight set consisting of one hundred corresponding bi-clusters. After this, consensus result is creating as follows:

Such algorithms results with eight bi-clusters, each with size $< |I^i|, |J^i| >$, where $|I^i|, |J^i|$ are cardinalities of clusters belonging to the i'th bi-cluster. Average AVC index is computed over all eight bi-cluster and its value is presented in table in rows marked as "Consensus" in "Type" column. For each method there is a corresponding row marked as "Normal". It contains average and best AVC value taken from all results of single method. To make a reliable comparison, "Normal" values has fixed cluster sizes set respectively to $\bar{I}$ *and* $\bar{J}$.

Table 4. Summary with average bi-cluster quality for text mining data [38].

Method	Type	Average AVC	Best AVC
PLSA	Normal	0.118	0.138
	Consensus	0.304	
K-L	Normal	0.129	0.147
	Consensus	0.297	
LSE	Normal	0.211	0.245
	Consensus	0.274	
nsK-L	Normal	0.140	0.154
	Consensus	0.253	
Cheng-Church	Normal	0,201	0,201
	Consensus	0,201	
BiMax	Normal	0,281	0,298
	Consensus	0,304	
CPB	Normal	0,164	0,187
	Consensus	0,192	
FABIA	Normal	0,288	0,299
	Consensus	0,320	
XMotifs	Normal	0,312	0,315
	Consensus	0,325	
ISA	Normal	0,402	0,421
	Consensus	0,452	
Qubic	Normal	0,187	0,207
	Consensus	0,159	
All results	Normal	0,221	0,421
	Consensus	0.385	

Table 5. Summary with average bi-cluster quality for microarray data [49].

Method	Type	Average AVC	Best AVC
PLSA	Normal	0.118	0.138
	Consensus	0.304	
K-L	Normal	0.129	0.147
	Consensus	0.297	
LSE	Normal	0.211	0.245
	Consensus	0.274	
nsK-L	Normal	0.140	0.154
	Consensus	0.253	
Cheng-Church	Normal	0,198	0,198
	Consensus	0,198	
BiMax	Normal	0,305	0,337
	Consensus	0,541	
CPB	Normal	0,158	0,185
	Consensus	0,198	

FABIA	Normal	0,327	0,363
	Consensus	0,429	
XMotifs	Normal	0,352	0,402
	Consensus	0,452	
ISA	Normal	0,502	0,596
	Consensus	0,602	
Qubic	Normal	0,187	0,207
	Consensus	0,228	
All results	Normal	0,238	0,402
	Consensus	0,391	

As clearly shown in the tables above, the algorithm based on combining the results of a wide variety of methods give much better results than individual algorithms. Resulting bi-cluster set, computed on all available data is in both cases almost twice time better than average score.

9.4.2.3. Other ways to determine the result quality.

In the case of real data, we do not know exactly what to expect. We do not have the expected bi-clusters, so we cannot use the same algorithm as in the case of synthetic data. Instead, we will analyze the results in terms of their meaningfulness.

Data contains genes annotated to one of eight gene ontology term from biological process ontology. Using Cytoscape program with its plugin BiNGO, we create a term ontology network based on a set of genes from every bi-cluster. In addition, check how found sets of genes are associated with the expected terms. In order to link set of terms with set of genes we using hypergeometric test with a significance level of 0.05.

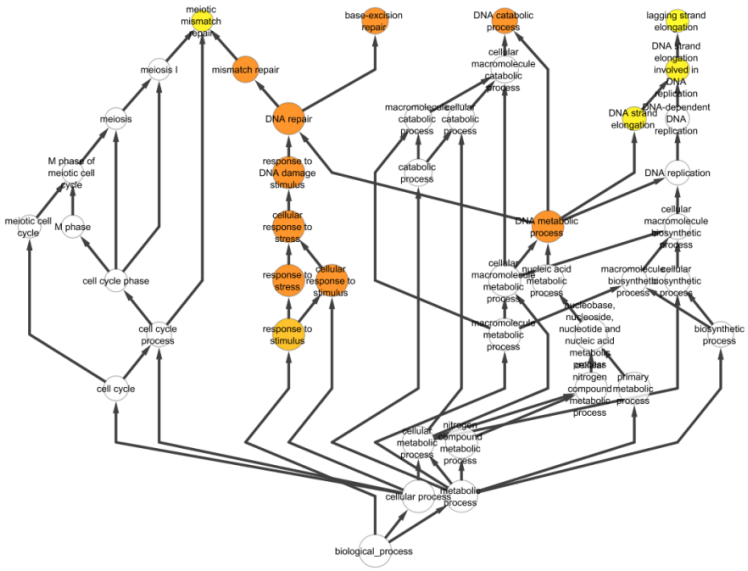


Figure 46. Sample network for gene cluster.

Figure 46 shows an example of a network generated for a single gene cluster by Kullback–Leibler algorithm. It consists of the terms that have been associated with this cluster using the BiNGO program (colored) and terms between them and the root (white). Terms are colored from yellow (the largest p-values) to the color orange (the smallest p-values). The attached image shows that terms for that cluster of genes are detected correctly because the terms are found mainly in the branch containing the expected term (with in this case is GO:0006950 “response to stress”).

To assess the quality of a single cluster, I chose two values. (1) The network density (higher means better). (2) Number of nodes in network - the lower means better. The data has been constructed in such a way that it contains eight bi-clusters, each of them described by one ontology term. Therefore, I decided that the best outcome is the one which creates a dense network focusing on that term.

Table 6. Comparison of gene ontology trees based on gene clusters.

Method	Network density	Numbers of nodes
PLSA	0,066	55,875
K-L	0,093	60,125
LSE	0,090	53,375
nsK-L	0,080	60,125
Consensus algorithm	0,139	34,375

10. Conclusions and summary

The aim of this thesis was to develop a universal approach to the analysis of bi-clustering and method that is resistant on the structure of the data. For this purpose, the synthetic dataset that covered almost all relevant data variants was created. Obtained on their basis results showed that the approach proposed in the dissertation is clearly better than the available methods or no worse than the three best algorithms (for this specific data). A measure of the quality of the synthetic data was the arithmetic mean of the measure defining the coverage obtained bi-clusters in a set of expected bi-clusters and measures of determining the coverage of expected bi-clusters in a set of found bi-clusters. In other words arithmetic mean of relevance and recovery.

The proposed method has also shown that it can improve performance for real data. For this purpose, analyzed two completely different sets of data available in the literature. It has been shown that this approach significantly improves the quality of the bi-clusters.

To confirm the described above thesis, were created synthetic data (described in Chapter 9.3.1) and selected from the literature, two sets of real data (described in Chapter 9.3.2). For this data set analysis were performed and discussed consecutively in Chapter 9.4.1, and 9.4.2. Both studies showed significant improvement in the quality of the results after applying the proposed method.

The result of work on the algorithm was universal and expanded software for analysis of bi-clustering. The software has been released to the public on the Internet, along with extensive service organized in the form of a blog. At the address http://aspectanalyzer.foszner.pl was posted ready to use installer, along with a complete user manual. In addition, the portal allows report bugs, new features, and questions about the software. Will be published also detailed information about current and planned versions. All software is provided free of charge and will include a complete, ready-to-run package.

Original added values of dissertation are:

- Developed similarity measures between bi-clusters
- The methodology of combining bi-clustering results based on generalized Hungarian algorithm,
- Meta-algorithm of bi-clustering combining the results of different methods
- The publicly available software with friendly graphical user interface

Bibliography

[1] J. N. Morgan and J. A. Sonquist, "Problems in the analysis of the survey data, and a proposal," *JAm Stat Assoc,* pp. 415-434, 1963.

[2] J. N. Hartigan, "Direct clustering of a data matrix," *JAm Stat Assoc,* pp. 123-129, 1972.

[3] B. Mirkin, "Mathematical Classification and Clustering," *Dordrecht: Kluwer.,* 1996.

[4] Y. Cheng and G. M. Church, "Biclustering of expression data," *In Proc. ISMB'00,* pp. 93-103, 2000.

[5] G. Li, Q. Ma, H. Tang, A. H. Paterson and Y. Xu, "QUBIC: a qualitative biclustering algorithm for analyses of gene expression data.," *Nucleic Acids Res.,* 2009.

[6] A. Prelic, S. Bleuler and P. Zimmermann, "A systematic comparison and evaluation of biclustering methods for gene expression data," *Bioinformatics,* p. 1122–9, 2006;.

[7] G. Getz, E. Levine and E. Domany, "Coupled two-way clustering analysis of gene microarray data," *In Proceedings of the Natural Academy of Sciences,* p. 12079–12084, 2000.

[8] D. Lee and S. Seung, "Algorithms for Non-negative Matrix Factorization," *Advances in neural information processing systems,* pp. 556-562, 2000.

[9] P. Foszner, A. Gruca and J. Polańska, "Distant Analysis of the GENEPI-ENTB Databank – System Overview," *Computer Networks, 17th Conference, CN 2010, Ustroń, Poland,* pp. 245-252, 15-19 czerwiec 2010.

[10] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. H. Witten, "The WEKA data mining software: an update," *ACM SIGKDD explorations newsletter,* pp. 10-18, 2009.

[11] P. Foszner, R. Jaksik, A. Gruca, J. Polańska and A. Polański, "Efficient reannotation system for verifying genomic targets of DNA microarray probes," *8th European Conference on Mathematical and Theoretical Biology, and Annual Meeting of The Society for Mathematical Biology,* czerwiec 28 – lipiec 2 2011.

[12] P. Foszner, A. Gruca and A. Polański, "Efficient system for clustering of dynamic document database," *Lectures Notes in Computer Science 6874,* pp. 186 -189, 2011.

[13] P. Foszner, A. Gruca and A. Polański, "Comparisons of biclustering algorithms," *IV Zjazd Polskiego Towarzystwa Bioinformatycznego połączony z 9. Warsztatami z Bioinformatyki dla Doktorantów,* 2011.

[14] P. Foszner, A. Gruca and A. Polański, "Distributed system for computing bi-clustering algorithms," *V Zjazd Polskiego Towarzystwa Bioinformatycznego połączony z 10. Warsztatami z Bioinformatyki dla Doktorantów,* 2012.

[15] S. C. Madeira and A. L. Oliveira, "Biclustering algorithms for biological data analysis: a survey," *IEEE/ACMTrans Comput Biol Bioinformatics,* pp. 24-45, 2004.

[16] R. Tibshirani, T. Hastie, M. Eisen, D. Ross, D. Botstein and P. Brown, "Clustering methods for the analysis of DNA microarray data," *Technical report, Department of Health Research and Policy, Department of Genetics and Department of Biochemestry, Stanford University,,* 1999.

[17] S. Busygin, G. Jacobsen and E. Kramer, "Double conjugated clustering applied o leukemia microarray data," *In Proceedings of the 2nd SIAM International Conference on Data Mining, Workshop on Clustering High Dimensional Data,* 2002.

[18] K. Eren, M. Deveci, O. Kucuktunc and U. V. Catalyurek, "A comparative analysis of biclustering algorithms for gene expression data," *BRIEFINGS IN BIOINFORMATICS,* pp. 279-292, 2012.

[19] D. Bozdag, J. D. Parvin and U. V. Catalyurek, "A biclustering method to discover co-regulated genes using diverse gene expression datasets," *In: Proceedings 1st International Conference Bioinformatics and Computational Biology,* p. 151–163, 2009.

[20] L. Teng and L. Chan, "Discovering biclusters by iteratively sorting with weighted correlation coefficient in gene expression data," *J. Signal Process. Syst.,* p. 267–280, 2008.

[21] W. Ayadi, M. Elloumi and J. K. Hao, "A biclustering algorithm based on a bicluster enumeration tree: application to dna microarray data," *BioData Mining,* 2009.

[22] A. Pascual-Montano, J. M. Carazo, K. Kochi, D. Lehmann and R. D. Pascual-Marqui, "Non-smooth Non-Negative Matrix Factorization," *IEEE Trans on Pattern Analysis and Machine Intelligence,* pp. 403-415, 2006.

[23] S. Hochreiter, U. Bodenhofer and M. Heusel, "FABIA: factor analysis for bicluster acquisition," *Bioinformatics,* p. 1520–1527, 2010.

[24] A. Tanay, R. Sharan, M. Kupiec and R. Shamir, "Revealing modularity and organization in the yeast molecular network by integrated analysis of highly heterogeneous genomewide data," *Proc Natl Acad Sci,* p. 2981–6, 2004.

[25] L. Lazzeroni and A. Owen, "Plaid models for gene expression data," *Stat Sin,* pp. 61-86, 2000.

[26] H. W. Kuhn, "The Hungarian Method for the assignment problem," *Naval Research Logistics Quarterly,* p. 83–97, 1955.

[27] D. Konig, "Uber Graphen und ihre Anwendung auf Determinantentheorie und Mengenlehre," *Math. Ann.,* pp. 453-465, 1916.

[28] J. Egervary, "Matrixok kombinatorius tulajdonsagairol," *Mat. Fiz. Lapok,* pp. 16-28, 1931.

[29] J. Munkres, "Algorithms for the Assignment and Transportation Problems," *Journal of the Society for Industrial and Applied Mathematics,* p. 32–38, 1957.

[30] W. Pierskalla, "The multidimensional assignment problem," *Operations Research,* pp. 422-431, 1968.

[31] A. B. Poore, "Multidimensional assignment formulation of data association problems arising from multitarget and multisensor tracking," *Computation Optimization and Applications,* pp. 27-54, 1994.

[32] R. Murphey, P. Pardalos and L. Pitsoulis, "A greedy randomized adaptive search procedure for the multitarget multisensor tracking problem," *DIMACS Series,* pp. 277-302, 1998.

[33] J. F. Pusztaszeri, P. E. Rensing and T. M. Liebling, "Tracking elementary particles near their primary vertex: acombinatorial approach," *Journal of Global Optimization,* pp. 41-64, 1996.

[34] C. J. Veenman, E. A. Hendriks and M. J. Reinders, "fast and robust point tracking algorithm," *Proceedings of the Fifth IEEE International Conferenceon Image Processing,* pp. 653-657, 1998.

[35] R. E. Burkard and E. Çela, "Quadratic and three-dimensional assignment problems," *Annotated Bibliographies in Combinatorial Optimization,* pp. 373-392, 1997.

[36] R. E. Burkard and E. Çela, "Linear Assignment Problems and extensions," *Handbook of Combinatorial Optimization,* pp. 75-149, 1999.

[37] E. Çela, "Assignment problems," *Oxford University Press,* pp. 661-678, 2002.

[38] M. Chagoyen, P. Carmona-Saez, H. Shatkay, J. M. Carazo and A. Pascual-Montano, "Discovering semantic features in the literature: a foundation for building functional associations," *BMC Bioinformatics,* 2006.

[39] [Online]. Available:

http://bioinformatics.cs.vt.edu/~murali/software/biorithm/index.html.

[40] R. Santamaría, R. Therón and L. Quintales, "BicOverlapper: A tool for bicluster visualization," *Bioinformatics,* pp. 1212-1213, 2008.

[41] J. Heinrich, M. Burch, R. Seifert and D. Weiskopf, "BiCluster Viewer: A Visualization Tool for Analyzing Gene Expression Data," *SimTech Cluster of Excellence,* 2011.

[42] P. Shannon , A. Markiel , O. Ozier , N. S. Baliga, J. T. Wang , D. Ramage , N. Amin , B. Schwikowski and T. Ideker , "Cytoscape: a software environment for integrated models of biomolecular interaction networks," *Genome Research,* pp. 2498-2504, 2003.

[43] S. Maere, K. Heymans and M. Kuiper, "BiNGO: a Cytoscape plugin to assess overrepresentation of Gene Ontology categories in Biological Networks," *BIOINFORMATICS APPLICATIONS NOTE,* p. 3448–3449, 2005.

[44] H. Kestler, A. Müller, J. Kraus, M. Buchholz, T. Gress, H. Liu, D. Kane, B. Zeeberg and J. Weinstein, "VennMaster: Area-proportional Euler diagrams for functional GO analysis of microarrays," *BMC Bioinformatics,* 2008.

[45] C. Huttenhower, K. T. Mutungu and N. Indik, "Detailing regulatory networks through large scale data integration," *Bioinformatics,* p. 3267–3274, 2009.

[46] J. Gu and J. S. Liu, "Bayesian biclustering of gene expression data," *BMC Genomics,* 2008.

[47] I. GmbH. [Online]. Available: http://ilnumerics.net/.

[48] K. Spark-Jones, "A statistical interpretation of term specificity and its application in retrieval.," *Journal of Documentation,* pp. 11-21, 1972.

[49] E. J. Yeoh, M. E. Ross and S. A. Shurtleff, "Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling," *Cancer,* p. 133–143, 2002.

[50] T. Hofmann , "Unsupervised Learning by Probabilistic Latent Semantic Analysis," *Machine Learning Journal,* pp. 177-196, 2001.

[51] A. Ben-Dor, B. Chor and R. Karp, "Discovering local structure in gene expression data: the order-preserving submatrix problem," *Journal of Computational Biology,* p. 373–384, 2003.

[52] S. Bergmann, J. Ihmels and N. Barkai, "Iterative signature algorithm for the analysis of large-scale gene expression data," *Phys Rev E,* 2003.

[53] Y. Kluger, R. Basri and J. T. Chang, "Spectral biclustering of microarray data: coclustering genes and conditions," *Genome Res,* p. 703–716, 2003.

[54] T. M. Murali and S. Kasif, "Extracting conserved gene expression motifs from gene expression data," *Pacific Symposium of Biocomputing,* p. 77–88, 2003.

List of Symbols and Abbreviations

- BBC – Bayesian Bi-Clustering
- FABIA – Factor Analysis for Bi-cluster acquisition
- QUBIC – Qualitative Biclustering
- IDF – Inverse Document Frequency
- LAP – linear assignment problem
- MAP – multi assignment problem
- NMF – non-negative matrix factorization

Table of Figures

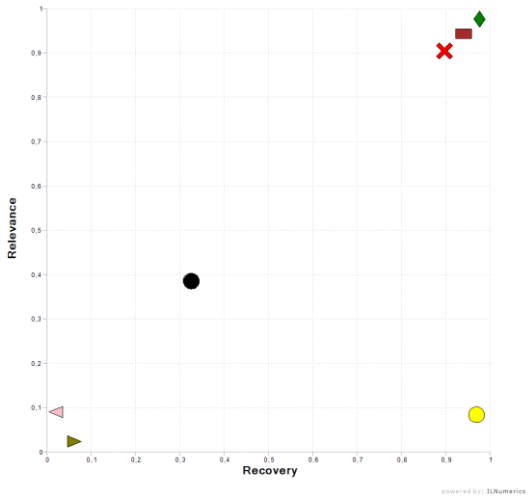
Index of tables

Appendix

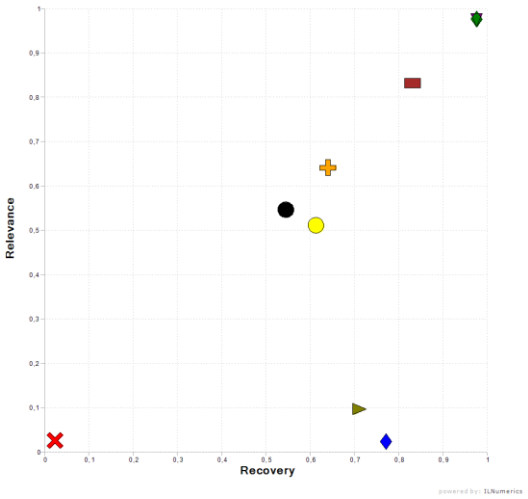
A. Synthetic data

Single bi-cluster

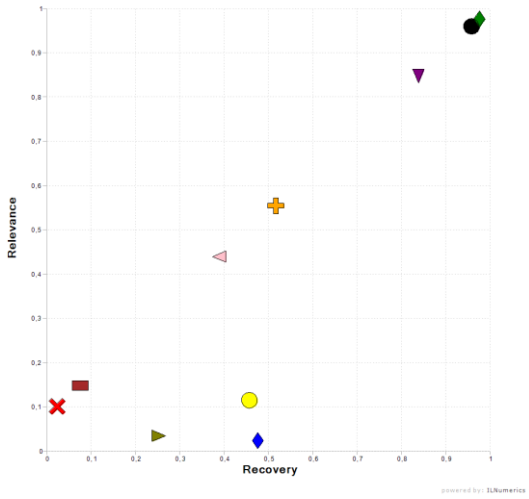
Constant data



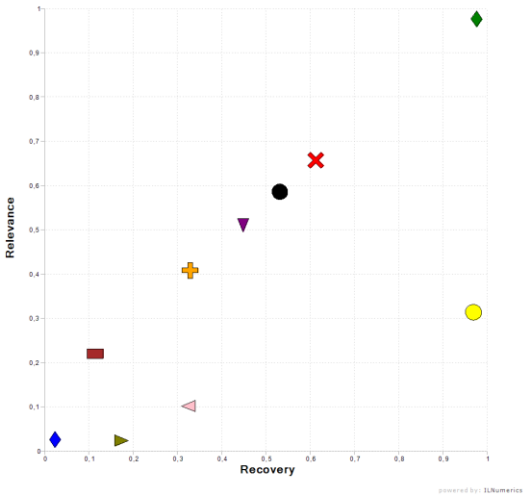
Constant data up-regulated



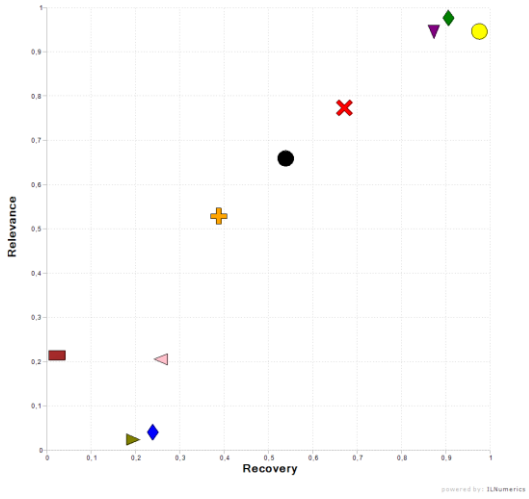
Plaid data



Shift-Scale data



Shift data



Scale data

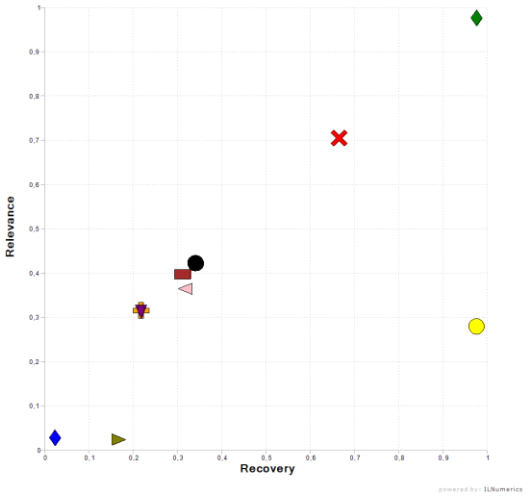


Table 7. Numeric results for single bi-cluster data with constant values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,385	0,385	0,385	1
Cheng-Church		0,926	0,926	0,926	1
BiMax		-	-	-	-
CPB		0,994	0,071	0,533	14,61
FABIA		-	-	-	-
XMotifs		0,966	0,966	0,966	1
Plaid		-	-	-	-
ISA		0,099	0,077	0,088	1,577
Qubic		0,133	0,008	0,071	16,83
Consensus		1	1	1	1

Table 8. Numeric results for single bi-cluster data with constant up-regulated values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,553	0,553	0,553	1
Cheng-Church		0,011	0,011	0,011	1
BiMax		0,788	0,009	0,398	86,5
CPB		0,623	0,516	0,57	75,86
FABIA		0,651	0,651	0,651	1
XMotifs		0,85	0,85	0,85	1
Plaid		1	1	1	1
ISA		-	-	-	-
Qubic		0,723	0,085	0,404	17,37
Consensus		1	1	1	1

Table 9. Numeric results for single bi-cluster data with plaid values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,983	0,983	0,983	1
Cheng-Church		0,083	0,083	0,083	1
BiMax		0,519	0,003	0,261	185
CPB		0,501	0,098	0,299	5,84
FABIA		0,558	0,558	0,558	1
XMotifs		0,133	0,133	0,133	1
Plaid		0,868	0,868	0,868	1
ISA		0,438	0,438	0,438	1
Qubic		0,302	0,014	0,158	23,67
Consensus		1	1	1	1

Table 10. Numeric results for single bi-cluster data with shift and scale values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,594	0,594	0,594	1
Cheng-Church		0,669	0,669	0,669	1
BiMax		0,131	0,013	0,072	10
CPB		0,994	0,312	0,653	1
FABIA		0,41	0,41	0,41	4,12
XMotifs		0,214	0,214	0,214	1
Plaid		0,519	0,519	0,519	1
ISA		0,409	0,091	0,25	4,57
Qubic		0,266	0,011	0,138	25,49
Consensus		1	1	1	1

Table 11. Numeric results for single bi-cluster data with shift values.

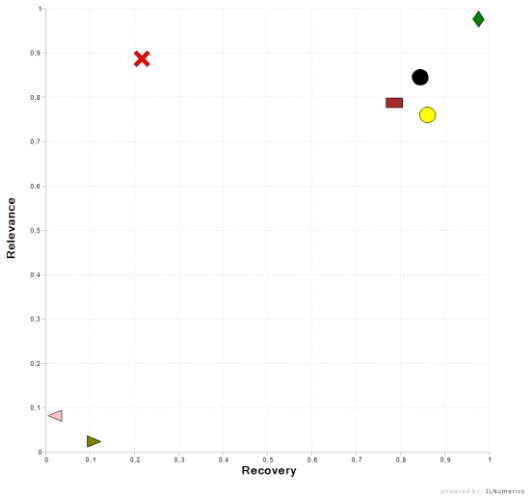
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,633	0,633	0,633	1
Cheng-Church		0,744	0,744	0,744	1
BiMax		0,381	0,032	0,206	25,96
CPB		1	0,912	0,956	1,19
FABIA		0,506	0,506	0,506	1
XMotifs		0,2	0,2	0,2	1
Plaid		0,914	0,914	0,914	1
ISA		0,399	0,191	0,295	2,12
Qubic		0,342	0,015	0,179	22,15
Consensus		0,941	0,941	0,941	1

Table 12. Numeric results for single bi-cluster data with scaled values.

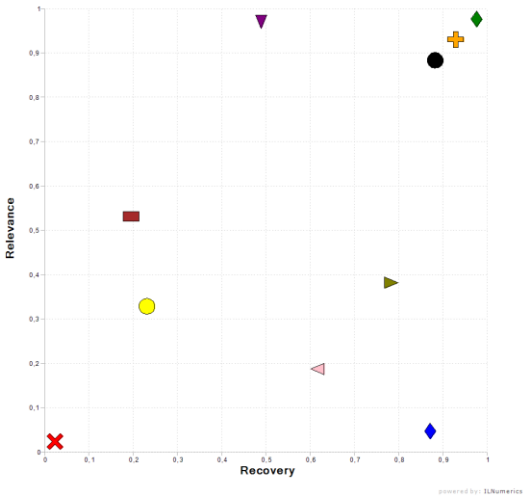
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,425	0,425	0,425	1
Cheng-Church		0,719	0,719	0,719	1
BiMax		0,137	0,015	0,076	8,969
CPB		1	0,277	0,638	4,39
FABIA		0,314	0,314	0,314	1
XMotifs		0,399	0,399	0,399	1
Plaid		0,314	0,314	0,314	1
ISA		0,406	0,365	0,385	1,24
Qubic		0,266	0,011	0,138	24,43
Consensus		1	1	1	1

Bi-clusters with exclusive rows and columns

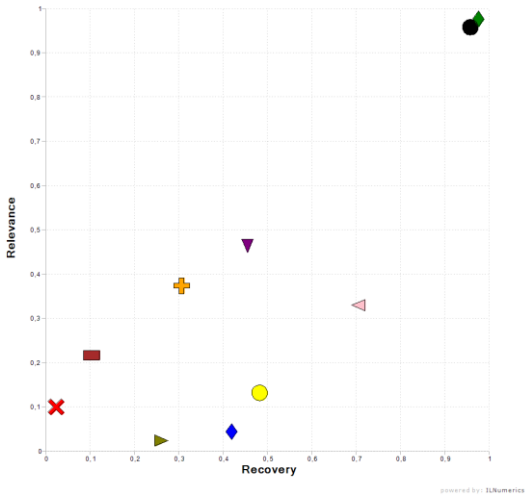
Constant data



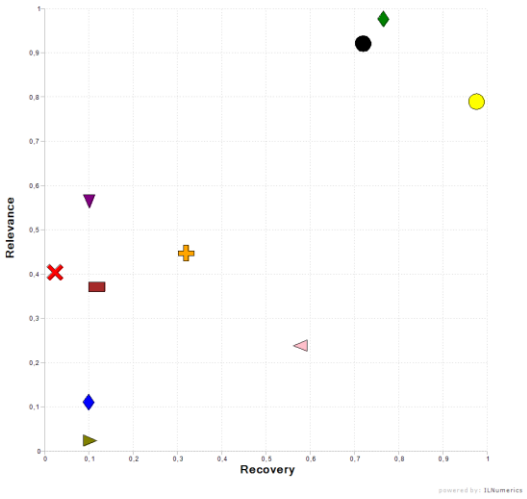
Constant data up-regulated



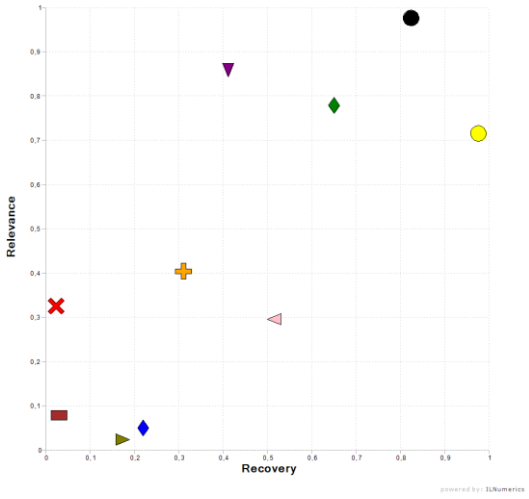
Plaid data



Shift-Scale data



Shift data



Scale data

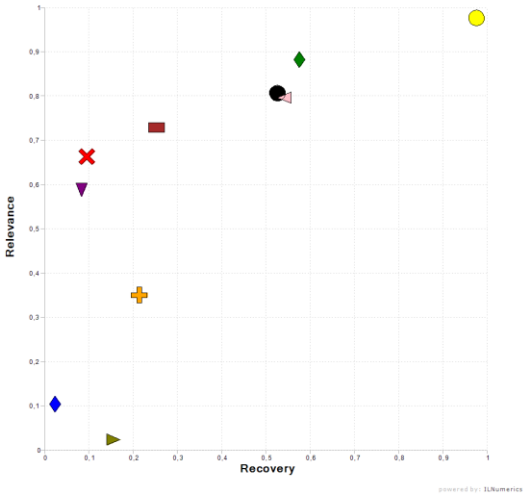


Table 13. Numeric results for exclusive row and columns data with constant values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,693	0,693	0,693	4
Cheng-Church		0,182	0,727	0,454	4
BiMax		-	-	-	-
CPB		0,706	0,623	0,665	5,4
FABIA		-	-	-	-
XMotifs		0,645	0,645	0,645	4
Plaid		-	-	-	-
ISA		0,024	0,069	0,047	4,078
Qubic		0,092	0,022	0,057	17,57
Consensus		0,8	0,8	0,8	4

Table 14. Numeric results for exclusive row and columns data with constant up-regulated values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,902	0,906	0,904	1
Cheng-Church		0,008	0,032	0,02	1
BiMax		0,891	0,056	0,473	64
CPB		0,224	0,342	0,283	125,36
FABIA		0,951	0,953	0,952	4
XMotifs		0,187	0,548	0,367	4
Plaid		0,493	0,997	0,745	4
ISA		0,628	0,198	0,413	12,89
Qubic		0,797	0,396	0,596	8,83
Consensus		1	1	1	4

Table 15. Numeric results for exclusive row and columns data with plaid values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,981	0,981	0,981	4
Cheng-Church		0,027	0,108	0,067	4
BiMax		0,431	0,051	0,241	52,18
CPB		0,496	0,14	0,318	14,72
FABIA		0,316	0,386	0,351	4
XMotifs		0,109	0,227	0,168	4
Plaid		0,468	0,48	0,474	4
ISA		0,726	0,342	0,534	8,56
Qubic		0,267	0,03	0,148	40,3
Consensus		1	1	1	4

Table 16. Numeric results for exclusive row and columns data with shift and scale values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,638	0,638	0,638	4
Cheng-Church		0,072	0,289	0,181	4
BiMax		0,134	0,092	0,113	5,92
CPB		0,846	0,549	0,698	6,53
FABIA		0,313	0,319	0,316	4
XMotifs		0,149	0,267	0,208	4
Plaid		0,135	0,399	0,267	4
ISA		0,525	0,178	0,351	11,92
Qubic		0,135	0,033	0,084	16,39
Consensus		0,675	0,675	0,675	4

Table 17. Numeric results for exclusive row and columns data with shift values.

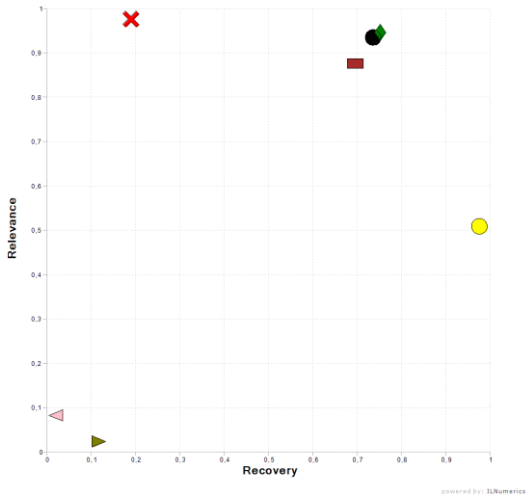
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,843	0,843	0,843	4
Cheng-Church		0,073	0,291	0,182	4
BiMax		0,261	0,057	0,159	18,68
CPB		0,989	0,622	0,806	6,71
FABIA		0,349	0,357	0,353	4
XMotifs		0,079	0,081	0,08	4
Plaid		0,446	0,746	0,596	4
ISA		0,548	0,266	0,407	8,34
Qubic		0,215	0,035	0,125	24,77
Consensus		0,676	0,676	0,676	0,676

Table 18. Numeric results for exclusive row and columns data with scaled values.

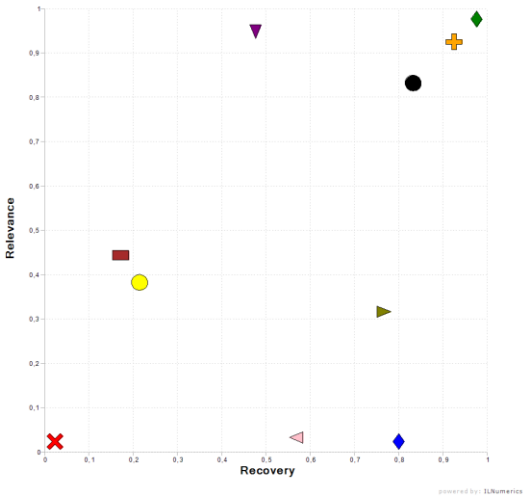
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,501	0,501	0,501	4
Cheng-Church		0,104	0,415	0,259	4
BiMax		0,038	0,077	0,057	4
CPB		0,916	0,604	0,76	6,3
FABIA		0,213	0,225	0,219	4
XMotifs		0,249	0,454	0,351	4
Plaid		0,093	0,371	0,232	4
ISA		0,518	0,495	0,506	4,24
Qubic		0,157	0,028	0,093	22,68
Consensus		0,547	0,547	0,547	4

Exclusive on rows and overlapping on columns (25%)

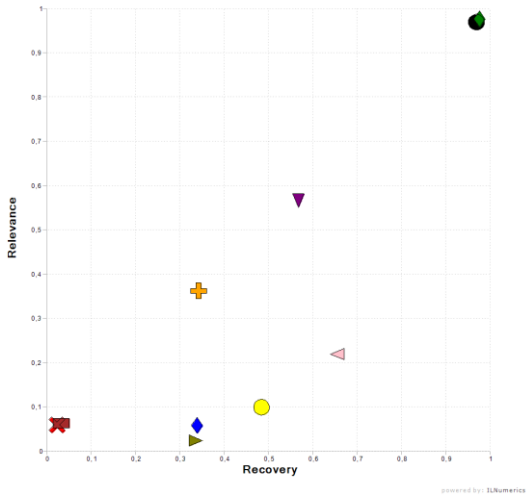
Constant data



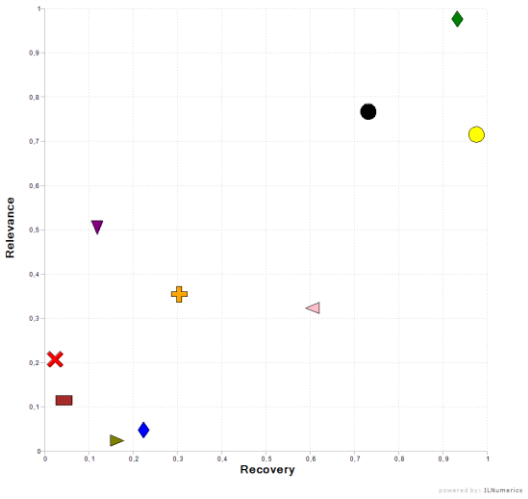
Constant data up-regulated



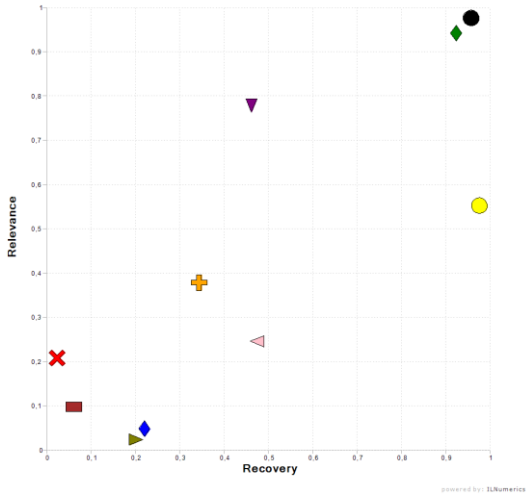
Plaid data



Shift-Scale data



Shift data



Scale data

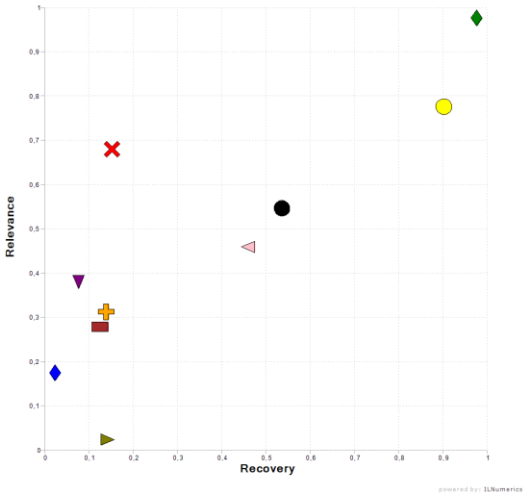


Table 19. Numeric results for single bi-cluster data with constant values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,663	0,669	0,666	4
Cheng-Church		0,175	0,699	0,437	4
BiMax		-	-	-	-
CPB		0,878	0,367	0,622	10,4
FABIA		-	-	-	-
XMotifs		0,628	0,628	0,628	4
Plaid		-	-	-	-
ISA		0,026	0,064	0,04	4,057
Qubic		0,108	0,022	0,065	19,86
Consensus		0,678	0,678	0,678	4

Table 20. Numeric results for single bi-cluster data with constant up-regulated values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,85	0,852	0,851	4
Cheng-Church		0,005	0,021	0,013	4
BiMax		0,816	0,021	0,419	154
CPB		0,205	0,39	0,297	66,08
FABIA		0,947	0,947	0,947	4
XMotifs		0,161	0,452	0,30	4
Plaid		0,478	0,974	0,726	4
ISA		0,577	0,03	0,304	76,48
Qubic		0,779	0,322	0,55	10,62
Consensus		1	1	1	4

Table 21. Numeric results for single bi-cluster data with plaid values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,993	0,993	0,993	4
Cheng-Church		0,022	0,09	0,056	4
BiMax		0,346	0,088	0,217	90
CPB		0,495	0,13	0,313	16,01
FABIA		0,35	0,39	0,37	4
XMotifs		0,032	0,093	0,063	4
Plaid		0,581	0,595	0,588	4
ISA		0,674	0,248	0,461	10,91
Qubic		0,34	0,055	0,198	28,42
Consensus		1	1	1	4

Table 22. Numeric results for single bi-cluster data with shift and scale values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,62	0,62	0,62	4
Cheng-Church		0,046	0,183	0,114	4
BiMax		0,208	0,058	0,133	14,5
CPB		0,819	0,579	0,699	6,07
FABIA		0,273	0,298	0,286	4
XMotifs		0,062	0,11	0,086	4
Plaid		0,123	0,417	0,27	4
ISA		0,52	0,273	0,396	7,68
Qubic		0,158	0,039	0,098	16,52
Consensus		0,783	0,783	0,783	4

Table 23. Numeric results for single bi-cluster data with shift values.

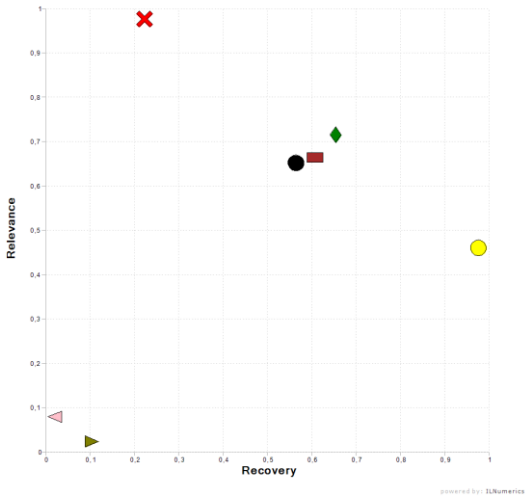
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,976	0,976	0,976	4
Cheng-Church		0,054	0,216	0,135	4
BiMax		0,248	0,058	0,153	18,24
CPB		0,995	0,557	0,776	7,44
FABIA		0,37	0,385	0,377	4
XMotifs		0,091	0,107	0,099	4
Plaid		0,487	0,783	0,635	4
ISA		0,502	0,253	0,378	8,03
Qubic		0,227	0,034	0,13	27,88
Consensus		0,943	0,943	0,943	4

Table 24. Numeric results for single bi-cluster data with scaled values.

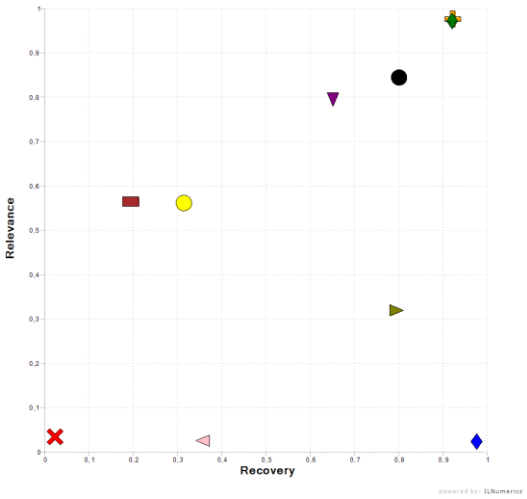
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,502	0,503	0,503	4
Cheng-Church		0,157	0,626	0,391	4
BiMax		0,041	0,162	0,102	4
CPB		0,832	0,714	0,773	5,1
FABIA		0,144	0,289	0,217	4
XMotifs		0,132	0,258	0,195	4
Plaid		0,088	0,353	0,221	4
ISA		0,436	0,424	0,43	4,14
Qubic		0,145	0,024	0,085	24,65
Consensus		0,898	0,898	0,898	4

Exclusive on rows and overlapping on columns (50%)

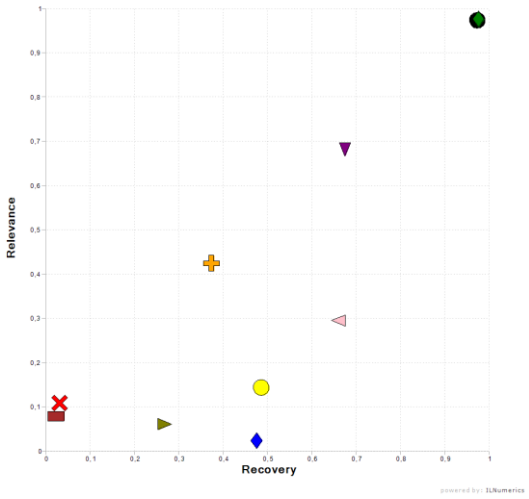
Constant data



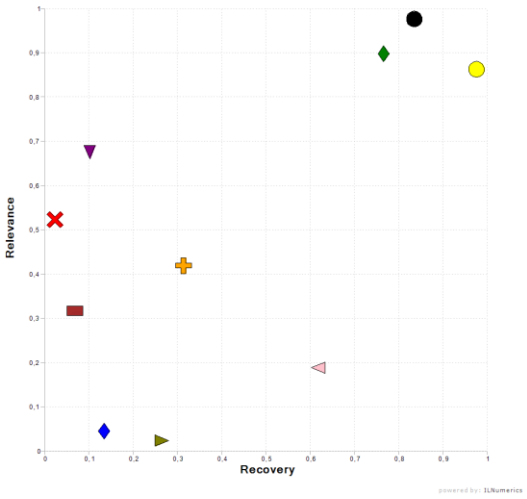
Constant data up-regulated



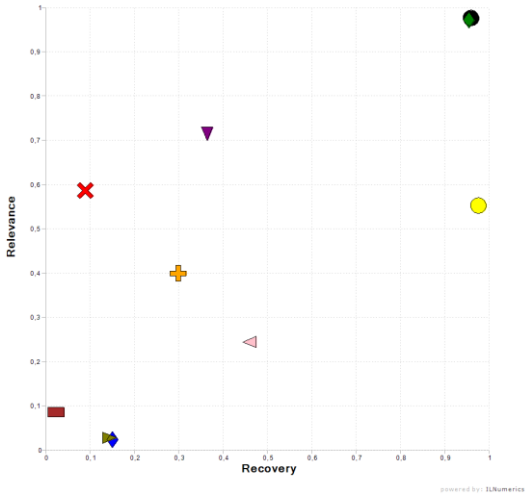
Plaid data



Shift-Scale data



Shift data



Scale data

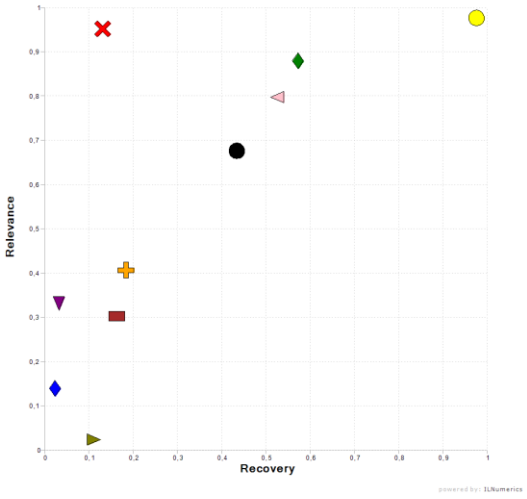


Table 25. Numeric results for single bi-cluster data with constant values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,518	0,547	0,532	4
Cheng-Church		0,205	0,819	0,512	4
BiMax		-	-	-	-
CPB		0,895	0,386	0,64	10,02
FABIA		-	-	-	-
XMotifs		0,557	0,557	0,557	4
Plaid		-	-	-	-
ISA		0,022	0,066	0,044	4,018
Qubic		0,093	0,019	0,056	20,17
Consensus		0,6	0,6	0,6	4

Table 26. Numeric results for single bi-cluster data with constant up-regulated values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,766	0,768	0,767	4
Cheng-Church		0,008	0,032	0,02	4
BiMax		0,938	0,022	0,48	167
CPB		0,292	0,511	0,401	57,33
FABIA		0,885	0,888	0,886	4
XMotifs		0,175	0,514	0,344	4
Plaid		0,621	0,725	0,673	4
ISA		0,337	0,024	0,18	57,14
Qubic		0,759	0,291	0,525	10,96
Consensus		0,884	0,884	0,884	4

Table 27. Numeric results for single bi-cluster data with plaid values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,997	0,997	0,997	4
Cheng-Church		0,024	0,098	0,061	4
BiMax		0,484	0,009	0,246	206
CPB		0,494	0,134	0,314	15,37
FABIA		0,378	0,426	0,402	4
XMotifs		0,017	0,067	0,042	4
Plaid		0,689	0,696	0,692	4
ISA		0,677	0,291	0,484	9,32
Qubic		0,267	0,048	0,157	26,51
Consensus		1	1	1	4

Table 28. Numeric results for single bi-cluster data with shift and scale values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,647	0,648	0,648	4
Cheng-Church		0,091	0,365	0,228	4
BiMax		0,167	0,067	0,117	10,58
CPB		0,743	0,578	0,66	5,63
FABIA		0,29	0,301	0,295	4
XMotifs		0,122	0,237	0,179	4
Plaid		0,145	0,462	0,304	4
ISA		0,5	0,157	0,328	12,84
Qubic		0,255	0,054	0,154	19,18
Consensus		0,599	0,599	0,599	4

Table 29. Numeric results for single bi-cluster data with shift values.

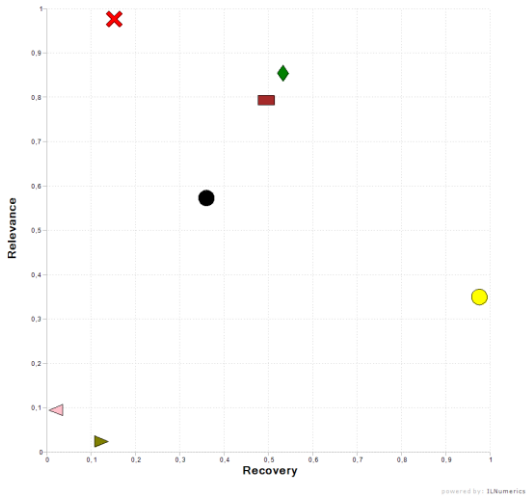
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,975	0,975	0,975	4
Cheng-Church		0,148	0,591	0,369	4
BiMax		0,206	0,036	0,121	23,2
CPB		0,991	0,557	0,774	7,46
FABIA		0,347	0,406	0,377	4
XMotifs		0,085	0,096	0,091	4
Plaid		0,41	0,72	0,565	4
ISA		0,503	0,253	0,378	8,09
Qubic		0,198	0,039	0,118	20,98
Consensus		0,97	0,97	0,97	4

Table 30. Numeric results for single bi-cluster data with scaled values.

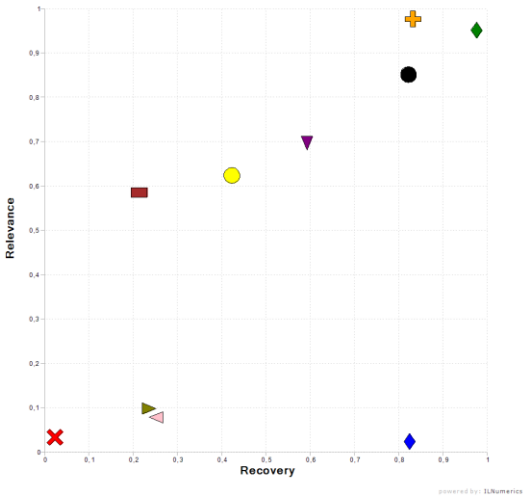
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,44	0,443	0,441	4
Cheng-Church		0,154	0,617	0,385	4
BiMax		0,052	0,105	0,078	4
CPB		0,952	0,632	0,792	6,39
FABIA		0,204	0,273	0,238	4
XMotifs		0,184	0,208	0,196	4
Plaid		0,061	0,227	0,144	4
ISA		0,529	0,519	0,524	4,09
Qubic		0,133	0,032	0,082	16,84
Consensus		0,571	0,571	0,571	4

Exclusive on rows and overlapping on columns (75%)

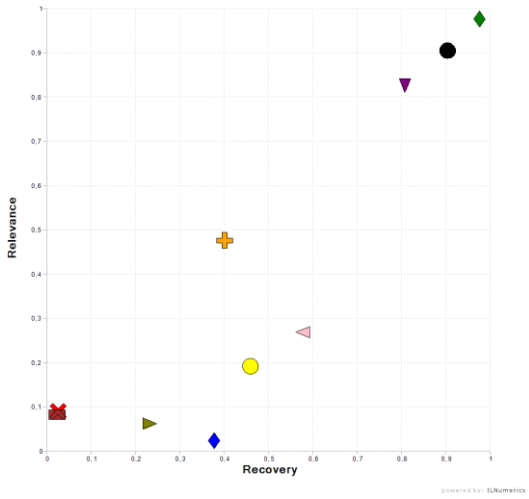
Constant data



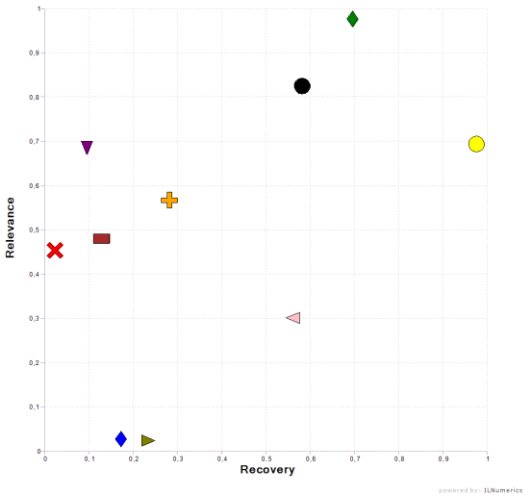
Constant data up-regulated



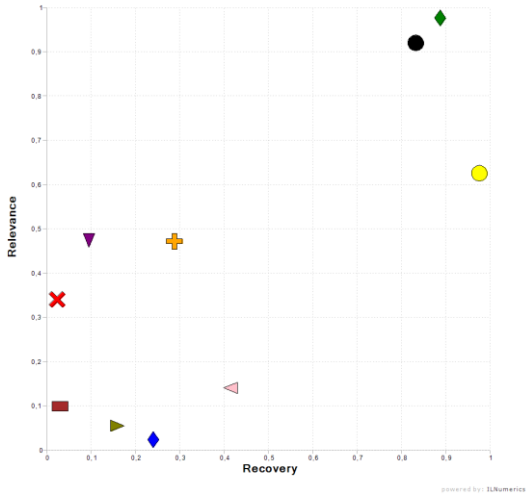
Plaid data



Shift-Scale data



Shift data



Scale data

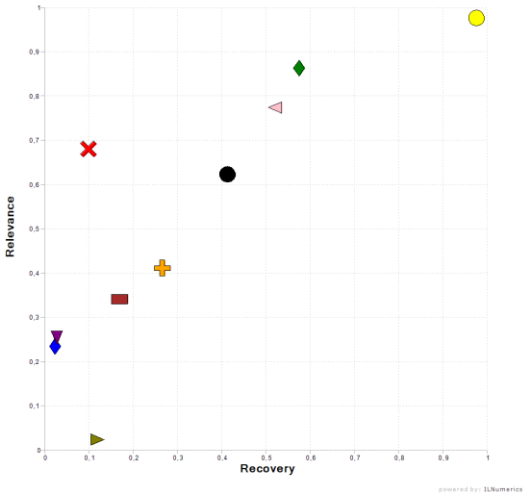


Table 31. Numeric results for single bi-cluster data with constant values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,335	0,335	0,335	4
Cheng-Church		0,141	0,565	0,353	4
BiMax		-	-	-	-
CPB		0,907	0,207	0,557	18,44
FABIA		-	-	-	-
XMotifs		0,46	0,46	0,46	4
Plaid		-	-	-	-
ISA		0,022	0,062	0,042	4,047
Qubic		0,112	0,021	0,067	21,25
Consensus		0,495	0,495	0,495	4

Table 32. Numeric results for single bi-cluster data with constant up-regulated values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,72	0,767	0,743	4
Cheng-Church		0,005	0,021	0,013	4
BiMax		0,722	0,013	0,368	225
CPB		0,363	0,559	0,461	30,69
FABIA		0,729	0,881	0,805	4
XMotifs		0,175	0,524	0,35	4
Plaid		0,515	0,628	0,571	4
ISA		0,212	0,062	0,137	14,52
Qubic		0,193	0,081	0,137	11,32
Consensus		0,858	0,858	0,858	4

Table 33. Numeric results for single bi-cluster data with plaid values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,926	0,926	0,926	4
Cheng-Church		0,019	0,075	0,047	4
BiMax		0,382	0,006	0,194	240
CPB		0,467	0,181	0,324	10,79
FABIA		0,407	0,478	0,442	4
XMotifs		0,017	0,067	0,042	4
Plaid		0,826	0,846	0,836	4
ISA		0,591	0,262	0,427	9,06
Qubic		0,23	0,046	0,138	24,02
Consensus		1	1	1	4

Table 34. Numeric results for single bi-cluster data with shift and scale values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,515	0,515	0,515	4
Cheng-Church		0,073	0,293	0,183	4
BiMax		0,191	0,038	0,115	24
CPB		0,827	0,437	0,632	8,18
FABIA		0,278	0,361	0,319	4
XMotifs		0,157	0,309	0,233	4
Plaid		0,13	0,433	0,282	4
ISA		0,501	0,202	0,351	10,12
Qubic		0,238	0,036	0,137	27,23
Consensus		0,606	0,606	0,606	4

Table 35. Numeric results for single bi-cluster data with shift values.

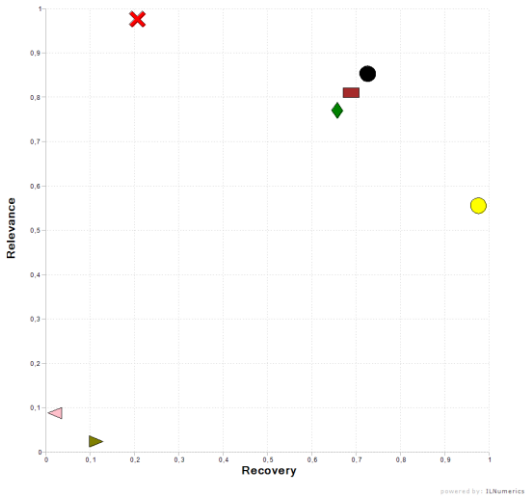
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,85	0,85	0,85	4
Cheng-Church		0,077	0,309	0,193	4
BiMax		0,284	0,014	0,149	85,5
CPB		0,987	0,575	0,781	7,28
FABIA		0,33	0,431	0,381	4
XMotifs		0,084	0,084	0,084	4
Plaid		0,145	0,436	0,291	4
ISA		0,453	0,123	0,288	15,03
Qubic		0,204	0,043	0,124	19,45
Consensus		0,903	0,903	0,903	4

Table 36. Numeric results for single bi-cluster data with scaled values.

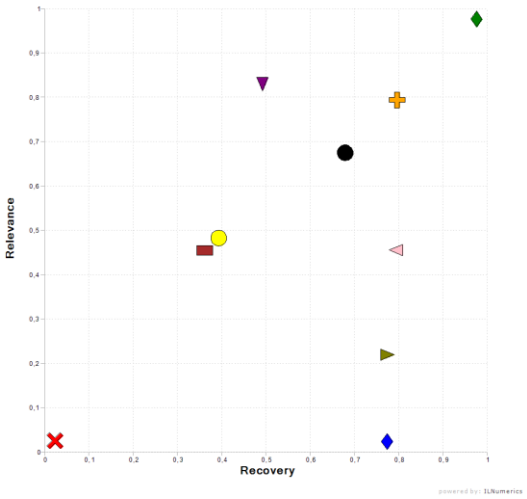
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,414	0,414	0,414	4
Cheng-Church		0,113	0,452	0,282	4
BiMax		0,041	0,162	0,102	4
CPB		0,955	0,643	0,799	6,28
FABIA		0,273	0,277	0,275	4
XMotifs		0,181	0,232	0,206	4
Plaid		0,044	0,177	0,111	4
ISA		0,521	0,512	0,517	4,08
Qubic		0,13	0,026	0,078	20,24
Consensus		0,57	0,57	0,57	4

Exclusive on columns and overlapping on rows (25%)

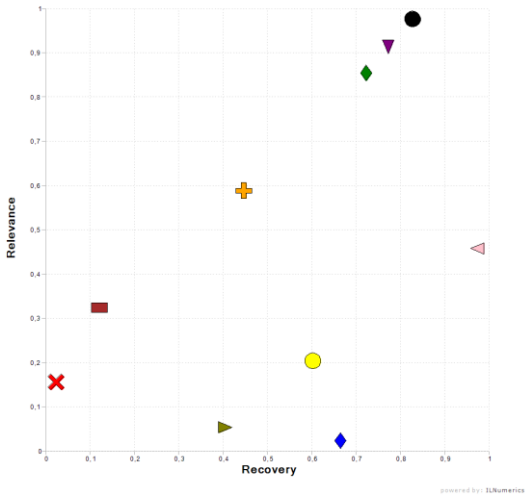
Constant data



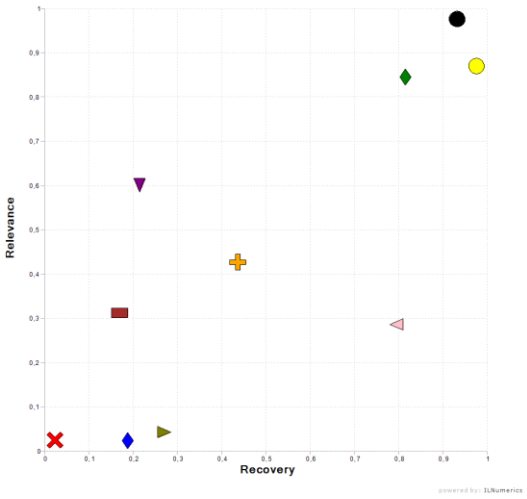
Constant data up-regulated



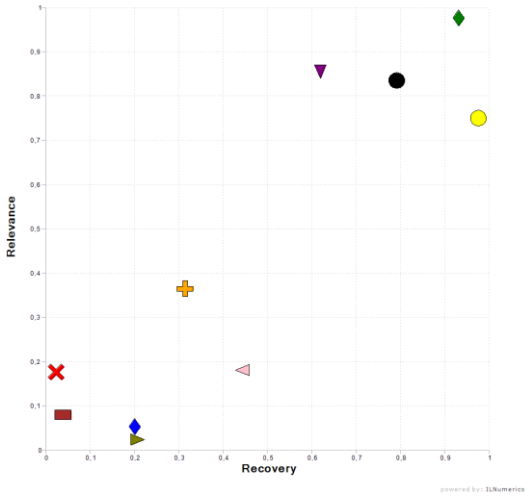
Plaid data



Shift-Scale data



Shift data



Scale data

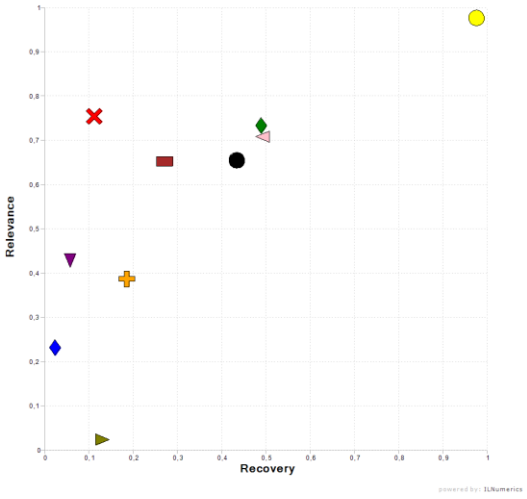


Table 37. Numeric results for single bi-cluster data with constant values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,638	0,64	0,639	4
Cheng-Church		0,183	0,732	0,458	4
BiMax		-	-	-	-
CPB		0,858	0,417	0,637	8,86
FABIA		-	-	-	-
XMotifs		0,606	0,608	0,607	4
Plaid		-	-	-	-
ISA		0,022	0,066	0,044	4
Qubic		0,099	0,018	0,059	22,43
Consensus		0,578	0,578	0,578	4

Table 38. Numeric results for single bi-cluster data with constant up-regulated values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,69	0,69	0,69	4
Cheng-Church		0,005	0,022	0,014	4
BiMax		0,789	0,02	0,404	159
CPB		0,392	0,492	0,442	7,24
FABIA		0,812	0,812	0,812	4
XMotifs		0,358	0,463	0,41	4
Plaid		0,495	0,851	0,673	4
ISA		0,812	0,464	0,638	7
Qubic		0,787	0,221	0,504	14,54
Consensus		1	1	1	4

Table 39. Numeric results for single bi-cluster data with plaid values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,601	0,601	0,601	4
Cheng-Church		0,024	0,096	0,06	4
BiMax		0,484	0,015	0,249	133
CPB		0,44	0,126	0,283	14,88
FABIA		0,328	0,363	0,345	4
XMotifs		0,094	0,2	0,147	4
Plaid		0,562	0,564	0,563	4
ISA		0,708	0,282	0,495	10,12
Qubic		0,296	0,033	0,165	39,05
Consensus		0,526	0,526	0,526	4

Table 40. Numeric results for single bi-cluster data with shift and scale values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,637	0,637	0,637	4
Cheng-Church		0,013	0,053	0,033	4
BiMax		0,126	0,052	0,089	9,65
CPB		0,667	0,572	0,619	5,2
FABIA		0,297	0,3	0,298	4
XMotifs		0,114	0,23	0,172	4
Plaid		0,144	0,408	0,276	4
ISA		0,544	0,213	0,379	10,26
Qubic		0,181	0,064	0,123	11,95
Consensus		0,557	0,557	0,557	4

Table 41. Numeric results for single bi-cluster data with shift values.

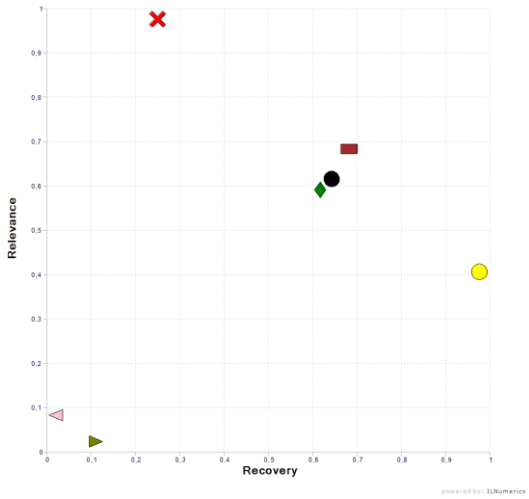
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,797	0,801	0,799	4
Cheng-Church		0,046	0,184	0,115	4
BiMax		0,219	0,069	0,144	13,22
CPB		0,977	0,722	0,849	5,63
FABIA		0,331	0,36	0,346	4
XMotifs		0,061	0,093	0,077	4
Plaid		0,629	0,823	0,726	4
ISA		0,459	0,188	0,324	9,92
Qubic		0,223	0,041	0,132	22,5
Consensus		0,934	0,934	0,934	4

Table 42. Numeric results for single bi-cluster data with scaled values.

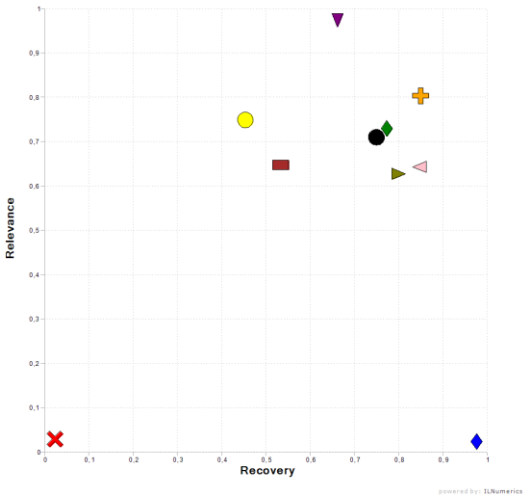
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,439	0,441	0,44	4
Cheng-Church		0,127	0,506	0,316	4
BiMax		0,041	0,162	0,102	4
CPB		0,964	0,652	0,808	6,17
FABIA		0,198	0,265	0,231	4
XMotifs		0,281	0,439	0,36	4
Plaid		0,073	0,294	0,184	4
ISA		0,498	0,476	0,487	4,24
Qubic		0,142	0,026	0,084	21,93
Consensus		0,492	0,492	0,492	4

Exclusive on columns and overlapping on rows (50%)

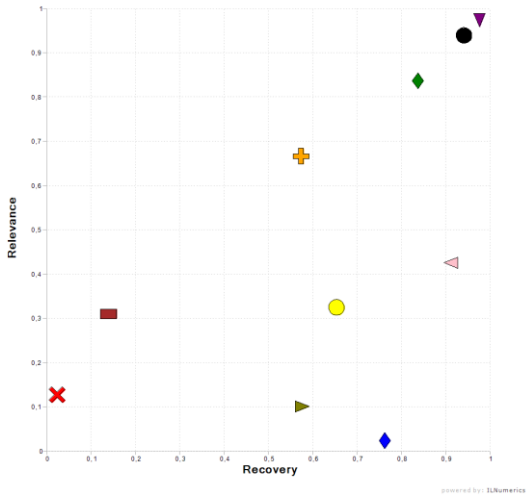
Constant data



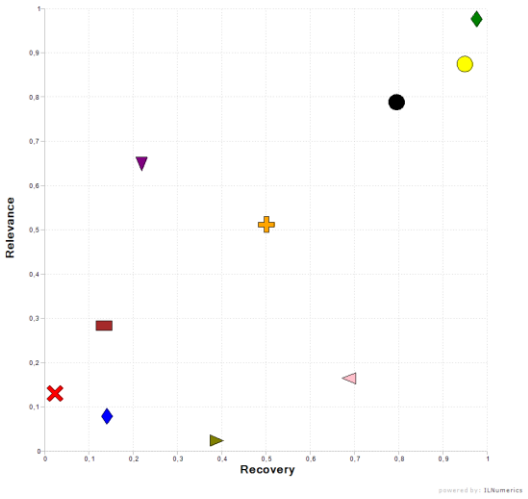
Constant data up-regulated



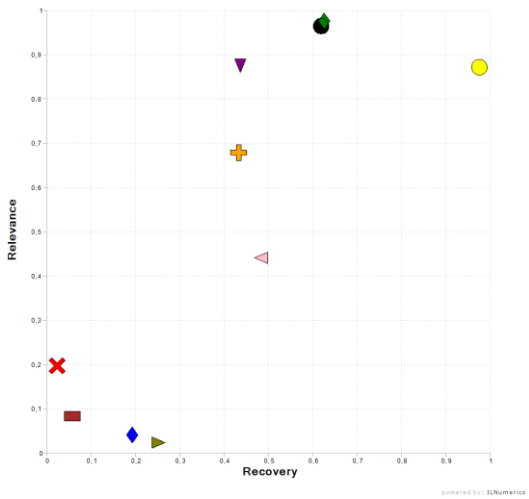
Plaid data



Shift-Scale data



Shift data



Scale data

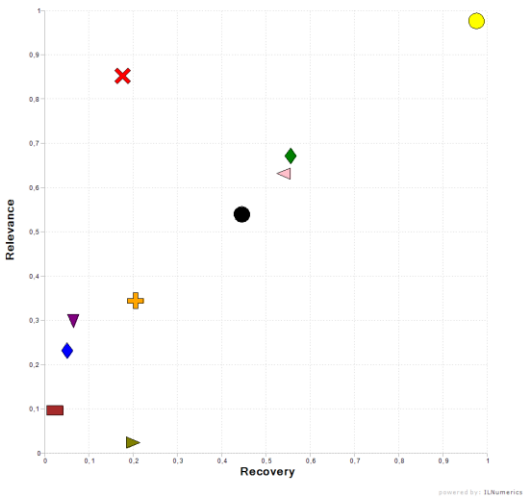


Table 43. Numeric results for single bi-cluster data with constant values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,529	0,529	0,529	4
Cheng-Church		0,21	0,839	0,524	4
BiMax		-	-	-	-
CPB		0,8	0,349	0,574	9,7
FABIA		-	-	-	-
XMotifs		0,561	0,587	0,574	4
Plaid		-	-	-	-
ISA		0,026	0,071	0,048	4
Qubic		0,095	0,02	0,057	19,63
Consensus		0,508	0,508	0,508	4

Table 44. Numeric results for single bi-cluster data with constant up-regulated values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,55	0,552	0,551	4
Cheng-Church		0,005	0,022	0,014	4
BiMax		0,72	0,017	0,369	165
CPB		0,328	0,583	0,455	15,88
FABIA		0,625	0,625	0,625	4
XMotifs		0,388	0,503	0,446	4
Plaid		0,484	0,76	0,622	4
ISA		0,625	0,5	0,562	5
Qubic		0,586	0,488	0,537	5,36
Consensus		0,568	0,568	0,568	4

Table 45. Numeric results for single bi-cluster data with plaid values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,6	0,6	0,6	4
Cheng-Church		0,02	0,081	0,05	4
BiMax		0,487	0,014	0,251	138
CPB		0,419	0,207	0,313	8,77
FABIA		0,368	0,425	0,397	4
XMotifs		0,094	0,197	0,145	4
Plaid		0,623	0,624	0,623	4
ISA		0,584	0,271	0,427	8,69
Qubic		0,368	0,063	0,216	24
Consensus		0,535	0,535	0,535	4

Table 46. Numeric results for single bi-cluster data with shift and scale values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,591	0,591	0,591	4
Cheng-Church		0,033	0,133	0,083	4
BiMax		0,118	0,097	0,108	4,98
CPB		0,702	0,65	0,676	4,93
FABIA		0,378	0,398	0,388	4
XMotifs		0,113	0,24	0,176	4
Plaid		0,174	0,495	0,335	4
ISA		0,515	0,157	0,336	13,3
Qubic		0,295	0,059	0,177	20,29
Consensus		0,721	0,721	0,721	4

Table 47. Numeric results for single bi-cluster data with shift values.

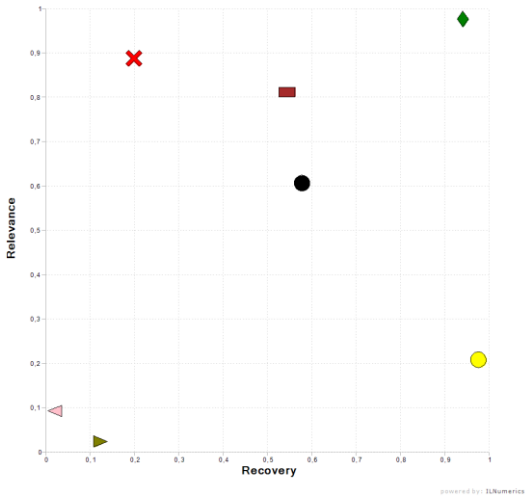
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,602	0,602	0,602	4
Cheng-Church		0,036	0,143	0,089	4
BiMax		0,197	0,05	0,123	15,91
CPB		0,941	0,546	0,744	7,26
FABIA		0,425	0,431	0,428	4
XMotifs		0,068	0,075	0,071	4
Plaid		0,428	0,55	0,489	4
ISA		0,476	0,288	0,382	6,76
Qubic		0,251	0,039	0,145	26,19
Consensus		0,608	0,608	0,608	4

Table 48. Numeric results for single bi-cluster data with scaled values.

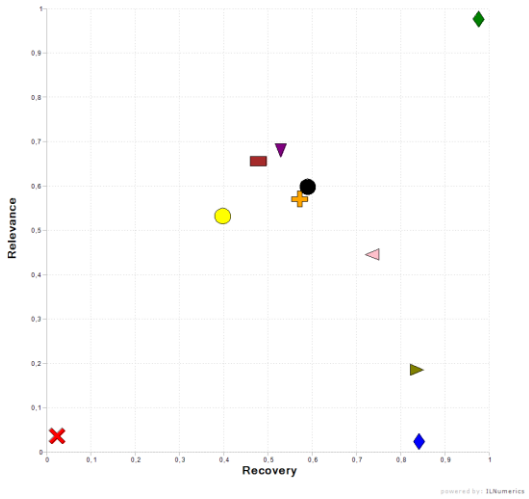
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,389	0,392	0,39	4
Cheng-Church		0,153	0,612	0,383	4
BiMax		0,044	0,175	0,109	4
CPB		0,852	0,7	0,776	5,31
FABIA		0,179	0,255	0,217	4
XMotifs		0,02	0,08	0,05	4
Plaid		0,056	0,224	0,14	4
ISA		0,474	0,457	0,465	4,18
Qubic		0,172	0,029	0,1	24,71
Consensus		0,485	0,485	0,485	4

Exclusive on columns and overlapping on rows (75%)

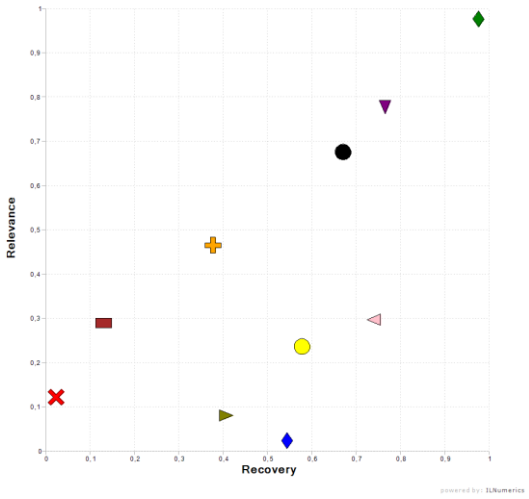
Constant data



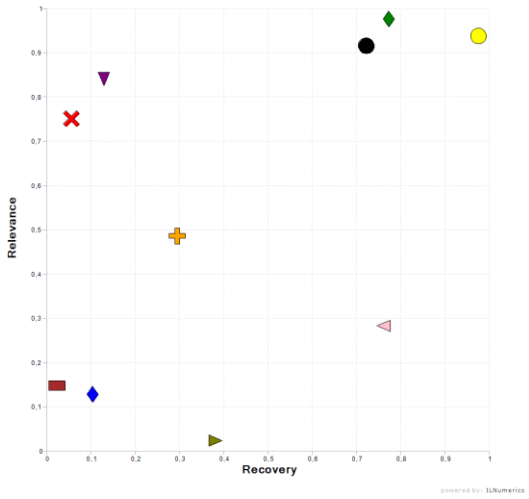
Constant data up-regulated



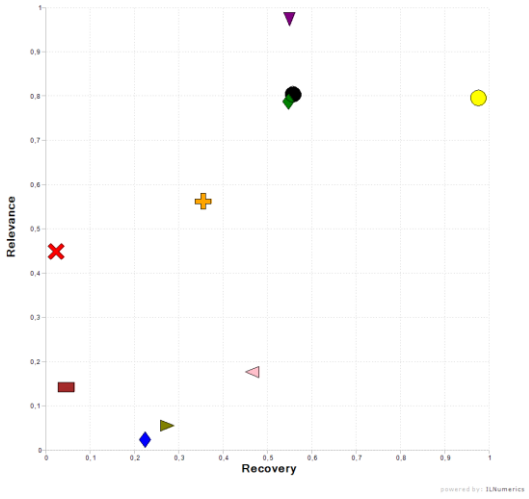
Plaid data



Shift-Scale data



Shift data



Scale data

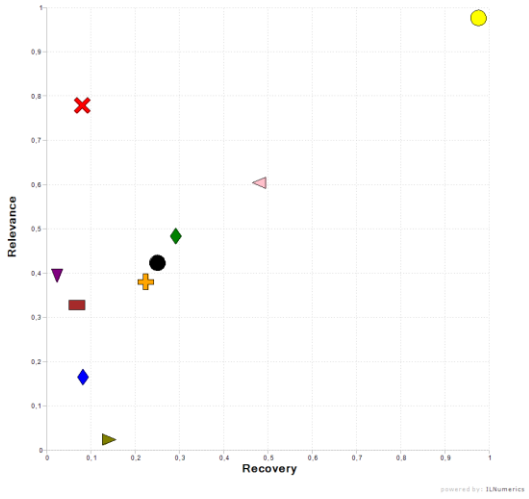


Table 49. Numeric results for single bi-cluster data with constant values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,414	0,415	0,414	4
Cheng-Church		0,151	0,606	0,379	4
BiMax		-	-	-	-
CPB		0,69	0,145	0,417	19,78
FABIA		-	-	-	-
XMotifs		0,391	0,554	0,472	4
Plaid		-	-	-	-
ISA		0,03	0,067	0,048	4,107
Qubic		0,097	0,02	0,059	19,76
Consensus		0,666	0,666	0,666	4

Table 50. Numeric results for single bi-cluster data with constant up-regulated values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,403	0,412	0,407	4
Cheng-Church		0,005	0,021	0,013	4
BiMax		0,58	0,013	0,296	184
CPB		0,268	0,366	0,317	36,63
FABIA		0,39	0,392	0,391	4
XMotifs		0,325	0,452	0,388	4
Plaid		0,36	0,47	0,415	4
ISA		0,507	0,305	0,406	6,82
Qubic		0,575	0,125	0,35	18,74
Consensus		0,674	0,674	0,674	4

Table 51. Numeric results for single bi-cluster data with plaid values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,482	0,482	0,482	4
Cheng-Church		0,02	0,081	0,051	4
BiMax		0,392	0,01	0,201	159
CPB		0,416	0,164	0,29	10,82
FABIA		0,273	0,33	0,301	4
XMotifs		0,097	0,202	0,149	4
Plaid		0,55	0,558	0,554	4
ISA		0,533	0,207	0,37	10,43
Qubic		0,292	0,051	0,171	24,81
Consensus		0,7	0,7	0,7	4

Table 52. Numeric results for single bi-cluster data with shift and scale values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,429	0,429	0,429	4
Cheng-Church		0,09	0,358	0,224	4
BiMax		0,114	0,091	0,103	5
CPB		0,558	0,439	0,498	8,62
FABIA		0,211	0,244	0,228	4
XMotifs		0,073	0,099	0,086	4
Plaid		0,127	0,398	0,262	4
ISA		0,45	0,157	0,304	11,65
Qubic		0,254	0,046	0,15	22,72
Consensus		0,455	0,455	0,455	4

Table 53. Numeric results for single bi-cluster data with shift values.

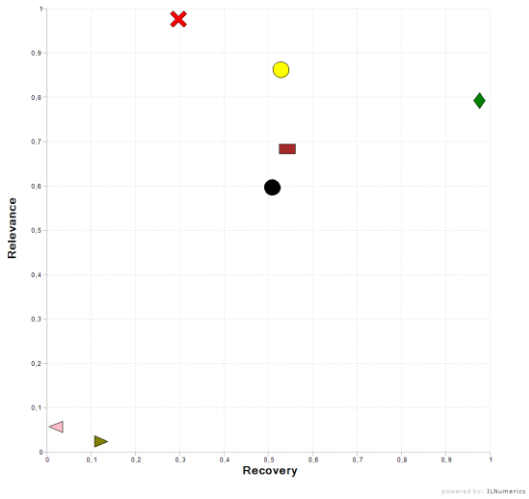
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,481	0,482	0,481	4
Cheng-Church		0,069	0,276	0,172	4
BiMax		0,223	0,029	0,126	31,78
CPB		0,803	0,477	0,64	7,19
FABIA		0,324	0,341	0,333	4
XMotifs		0,086	0,097	0,092	4
Plaid		0,474	0,582	0,528	4
ISA		0,411	0,117	0,264	14,21
Qubic		0,26	0,047	0,153	22,37
Consensus		0,473	0,473	0,473	4

Table 54. Numeric results for single bi-cluster data with scaled values.

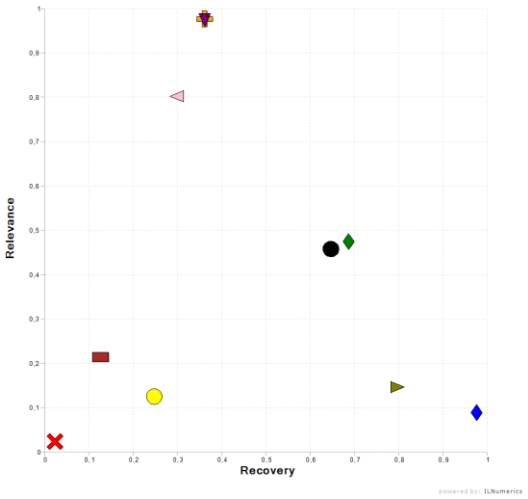
Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,279	0,281	0,28	4
Cheng-Church		0,123	0,494	0,309	4
BiMax		0,125	0,126	0,126	4
CPB		0,943	0,612	0,777	6,5
FABIA		0,255	0,255	0,255	4
XMotifs		0,113	0,224	0,168	4
Plaid		0,072	0,265	0,168	4
ISA		0,492	0,389	0,44	5,24
Qubic		0,177	0,042	0,11	17,58
Consensus		0,317	0,317	0,317	4

Overlapping on both (up to 100% of overlap)

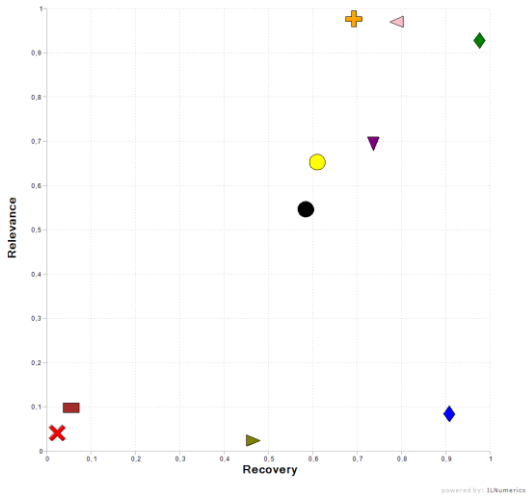
Constant data



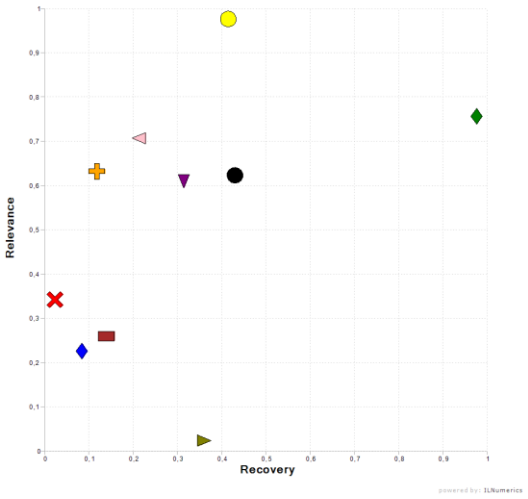
Constant data up-regulated



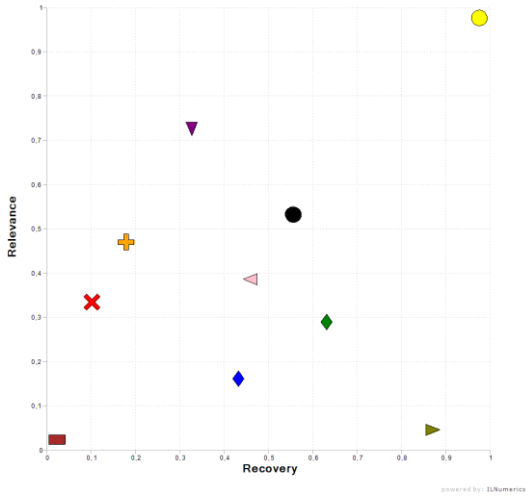
Plaid data



Shift-Scale data



Shift data



Scale data

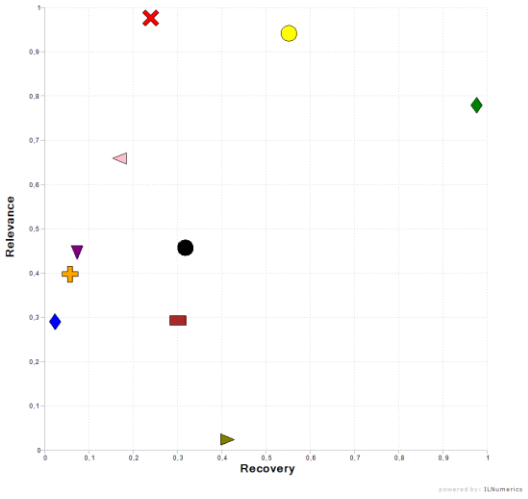


Table 55. Numeric results for single bi-cluster data with constant values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,377	0,545	0,461	4
Cheng-Church		0,222	0,886	0,554	4
BiMax		-	-	-	-
CPB		0,392	0,784	0,588	54,16
FABIA		-	-	-	-
XMotifs		0,402	0,623	0,513	4
Plaid		-	-	-	-
ISA		0,02	0,059	0,04	4,02
Qubic		0,091	0,029	0,06	13,69
Consensus		0,721	0,721	0,721	4

Table 56. Numeric results for single bi-cluster data with constant up-regulated values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,456	0,467	0,462	4
Cheng-Church		0,005	0,022	0,014	4
BiMax		0,694	0,089	0,391	31,34
CPB		0,168	0,126	0,147	243,76
FABIA		0,25	1	0,625	4
XMotifs		0,08	0,216	0,148	4
Plaid		0,25	1	0,625	4
ISA		0,206	0,821	0,514	4
Qubic		0,563	0,148	0,355	17,31
Consensus		0,485	0,485	0,485	4

Table 57. Numeric results for single bi-cluster data with plaid values

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,369	0,379	0,374	4
Cheng-Church		0,017	0,067	0,042	4
BiMax		0,572	0,093	0,333	36,19
CPB		0,385	0,445	0,415	7,55
FABIA		0,437	0,645	0,541	4
XMotifs		0,037	0,101	0,069	4
Plaid		0,464	0,472	0,468	4
ISA		0,499	0,641	0,57	4
Qubic		0,293	0,056	0,174	21,47
Consensus		0,615	0,615	0,615	4

Table 58. Numeric results for single bi-cluster data with shift and scale values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,248	0,417	0,332	4
Cheng-Church		0,063	0,25	0,157	4
BiMax		0,09	0,181	0,136	4
CPB		0,241	0,627	0,434	63,2
FABIA		0,106	0,422	0,264	4
XMotifs		0,115	0,201	0,158	4
Plaid		0,195	0,411	0,303	4
ISA		0,15	0,467	0,308	4,01
Qubic		0,215	0,061	0,138	14,39
Consensus		0,496	0,496	0,496	4

Table 59. Numeric results for single bi-cluster data with shift values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,248	0,483	0,365	4
Cheng-Church		0,079	0,314	0,196	4
BiMax		0,202	0,167	0,184	4,97
CPB		0,404	0,861	0,633	4,01
FABIA		0,107	0,429	0,268	4
XMotifs		0,049	0,049	0,049	4
Plaid		0,163	0,65	0,406	4
ISA		0,213	0,358	0,285	4,02
Qubic		0,364	0,068	0,216	22,03
Consensus		0,276	0,276	0,276	4

Table 60. Numeric results for single bi-cluster data with scaled values.

Method name	Chart symbol	Recovery	Relevance	Score	Average Num. of bi-clusters
BBC		0,204	0,331	0,267	4
Cheng-Church		0,165	0,659	0,412	4
BiMax		0,056	0,225	0,141	4
CPB		0,321	0,637	0,479	18,85
FABIA		0,073	0,293	0,183	4
XMotifs		0,195	0,226	0,211	4
Plaid		0,081	0,325	0,203	4
ISA		0,131	0,458	0,294	4
Qubic		0,25	0,057	0,153	18,21
Consensus		0,534	0,534	0,534	4