J Glob Optim (2007) 39:323–346
DOI 10.1007/s10898-007-9140-6

ORIGINAL PAPER

A novel clustering approach and prediction of optimal number of clusters: global optimum search with enhanced positioning

Meng Piao Tan · James R. Broach · Christodoulos A. Floudas

Received: 10 July 2006 / Accepted: 15 January 2007 / Published online: 12 April 2007

Abstract Cluster analysis of genome-wide expression data from DNA microarray hybridization studies is a useful tool for identifying biologically relevant gene groupings (DeRisi et al. 1997; Weiler et al. 1997). It is hence important to apply a rigorous yet intuitive clustering algorithm to uncover these genomic relationships. In this study, we describe a novel clustering algorithm framework based on a variant of the Generalized Benders Decomposition, denoted as the Global Optimum Search (Floudas et al. 1989; Floudas 1995), which includes a procedure to determine the optimal number of clusters to be used. The approach involves a pre-clustering of data points to define an initial number of clusters and the iterative solution of a Linear Programming problem (the primal problem) and a Mixed-Integer Linear Programming problem (the master problem), that are derived from a Mixed Integer Nonlinear Programming problem formulation. Badly placed data points are removed to form new clusters, thus ensuring tight groupings amongst the data points and incrementing the number of clusters until the optimum number is reached. We apply the proposed clustering algorithm to experimental DNA microarray data centered on the Ras signaling pathway in the yeast *Saccharomyces cerevisiae* and compare the results to that obtained with some commonly used clustering algorithms. Our algorithm compares favorably against these algorithms in the aspects of intra-cluster similarity and inter-cluster dissimilarity, often considered two key tenets of clustering. Furthermore, our algorithm can predict the optimal number of clusters, and the biological coherence of the predicted clusters is analyzed through gene ontology.

Keywords Clustering · Microarray data · Optimization

M. P. Tan · C. A. Floudas (✉)
Department of Chemical Engineering, Princeton University,
Princeton, NJ 08544, USA
e-mail: floudas@titan.princeton.edu

J. R. Broach
Department of Molecular Biology, Princeton University,
Princeton, NJ 08544, USA

1 Introduction

The aim of cluster analysis is to establish a set of clusters such that the data points in a cluster are more similar to one another than they are to those in other clusters. The clustering problem is old, can be traced back to Aristotle, and has already been studied quite extensively by 18th century naturalists such as Buffon, Cuvier, and Linne (Hansen and Jaumard 1997). Since then, clustering has been used in many disciplines, such as market research, social network analysis, and geology, thus reflecting its broad appeal and utility as a key step in exploratory data analysis (Jain et al. 1999). In market research for instance, cluster analysis is widely used when working with multivariate data from surveys and test panels. Market researchers use cluster analysis methods to segment and determine target markets, and position new products. Cluster analysis is also used in the service of market approaches to the establishment of business enterprise value. Johnson (2001) addresses the potential role and utility of cluster analysis in transfer pricing practices. Given the importance of clustering, a substantial number of books, such as Duran and Odell (1974), Hartigan (1975), and Jain and Dubes (1988), as well as review papers, such as Xu and Wunsch (2005) have been published on this subject.

In biology, clustering provides insights into transcriptional networks, physiological responses, gene identification, genome organization, and protein structure. Genome-wide measurements of mRNA expression levels have provided an efficient and comprehensive means of gathering information on genetic functions and transcriptional networks. However, extracting useful information from the resulting large data sets first involves organizing genes by their pattern and/or intensity of expression in order to define those genes that are co-regulated. Such information provides a basis for extracting regulatory motifs for transcription factors driving the diverse expression patterns, allowing assembly of predictive transcriptional networks (Beer and Tavazoie 2004). This information also provides insights into the functions of unknown genes, since functionally related genes are often co-regulated (Troyanskaya et al. 2003). Furthermore, clustered array data provides identification of distinct categories of otherwise indistinguishable cell types, which can have profound implications in processes such as disease progression (Sorlie et al. 2003). In sequence analysis, clustering is used to group homologous sequences into gene families. Examining characteristic DNA fragments helps in the identification of gene structures and reading frames. In protein structure prediction, clustering the ensemble of low energy conformers is used to identify the top suggested protein structures.

Two popular similarity metrics are correlation and Euclidean distance. The latter is often popular, since it is intuitive, can be described by a familiar distance function, and satisfies the triangular inequality. Clustering methods that employ asymmetric distance measures (Pipenbacher et al. 2002; Leisch et al. 1998) are probably more difficult to intuitively comprehend even though they may be highly suited to their intended applications. The earliest work on clustering emphasized visual interpretations for the ease of study, resulting in methods that utilize dendograms and color maps (Claverie 1999). Other examples of clustering algorithms include: (a) Single-Link and Complete-Link Hierarchical Clustering (Sokal and Michener 1958; Jain and Dubes 1988), (b) K-Means Algorithm and its family of variants, such as the K-Medians (Hartigan and Wong 1979; Zhang et al. 1999; Zhang 2000; Likas et al. 2003), (c) Reformulation Linearization-based Clustering (Sherali and Desai 2005a; Adams and Sherali 1990), (d) Fuzzy Clustering (Ruspini 1969; Dunn 1973; Bezdek 1981; Sherali and Desai

2005b), (e) Quality Cluster Algorithm (QTClust) (Heyer et al. 1999), (f) Graph-Theoretic Clustering (Zahn 1971; Wu and Leahy 1993; Gower and Ross 1969), (g) Mixture-Resolving Clustering Method (Dempster et al. 1977; Jain et al. 1999), (h) Mode Seeking Algorithms (Jain et al. 1999), (i) Artificial Neural Networks for Clustering (Kohonen 1984; Carpenter and Grossberg 1990) such as the Self-Organizing Map (SOM) (Kohonen 1997) and a variant that combines the SOM with hierarchical clustering, the Self-Organizing Tree Algorithm (SOTA) (Herrero et al. 2001) (j) Information-Based Clustering (Dhillon and Guan 2003; Tishby et al. 1999; Slonim et al. 2005), (k) Stochastic Approaches (Kirkpatrick et al. 1983; Metropolis et al. 1953; Lukashin and Fuchs 2001).

In this paper, we present a novel Mixed-Integer Nonlinear Programming (MINLP)-based clustering algorithm, the Global Optimal Search with Enhanced Positioning (EP_GOS_Clust), which is robust yet intuitive. This algorithm is significant in that it is able to progressively identify and weed out outlier data points. Also, our algorithm contains a convenient method to predict the optimal number of clusters. We compare our algorithm with several approaches commonly used in clustering biological microarray data, namely K-methods, QTClust., SOM, and SOTA. We use two assessment criteria to assess the results: the intra-cluster and inter-cluster error sums. We also examine the difference between the two error sums. In an optimal cluster configuration, the intra-cluster error sum is to be minimized and the inter-cluster error sum to be maximized. In this respect, we show that our proposed algorithm compares favorably. We also incorporate a methodology to predict the optimal number of clusters. In addition, in view of the context of the particular test dataset used, we compare the strength of biological coherence uncovered by the various approaches using Gene Ontology resources, and also the level of correlation between data points with the same cluster. We base our comparative study on actual DNA microarray data, though our algorithm can be readily utilized for data from other applications.

2 Methods

2.1 Experimental data

For the clustering studies described in this report, we used experimental microarray data derived from a study in the role of the Ras/protein kinase A pathway (PKA) on glucose signaling in yeast (Wang et al. 2004). These experiments analyzed mRNA levels in samples extracted from cells at various times following stimulation by glucose or following activation of either Ras2 or Gpa2, which are small GTPases involved in the metabolic and transcriptional response of yeast cells to glucose (Schneper et al. 2004). These experiments were performed in wild type cells and cells defective in PKA activity. Clustering these microarray data has proven to be a critical step in using the data to develop a predictive model of a topological map of the signaling network surrounding the Ras/PKA pathway (Lin et al. 2003).

Levels of RNA for each of the 6,237 yeast genes in each of the RNA samples from the above experiments were measured using Affymetrix microarray chips and analyzed by the Affymetrix software. Each of the eight test and control experiments consisted of four time points over a hour period, yielding 32 data points for each of the 6,237 genes. We used the Affymetrix MicroArray Suite 5.0, which analyzes the consensus of intensities of hybridization of an RNA to the collection of perfect match probes for a gene on the array, relative to the intensities of hybridization to single mismatch probes, to further determine whether a signal for a specific RNA in a sample was reliable (P or present), unreliably low (A or absent), or ambiguous (M). Before clustering the array data, we filtered the data to remove unreliable data. In particular, we retained all genes for which all the time points were present (4,105 genes), all the genes for which greater than 50% of the time points were present, and all the genes for which the present/absent calls exhibited a biologically relevant pattern (e.g. PAAA for the four time points in the experiment, suggested repression of gene expression over the course of the experiment). In all, we retained 5,652 genes. The expression patterns for these genes are then z-normalized over each gene.

2.2 Theoretical and computational framework

2.2.1 Notation

We denote the measure of distance for a gene i, for $i = 1, \dots, n$ having k features (or dimensions), for $k = 1, \dots, s$ as a_{ik}. Each gene's 32-time point expression pattern is transformed into a 24-dimensional vector, for which each vector element indicates the change in normalized expression level between time points for each gene, a_{ik}. Each gene is to be assigned to only one (hard clustering) of c possible clusters, each with center z_{jk}, for $j = 1, \dots, c$. The binary variables w_{ij} indicates whether gene i falls within cluster j ($w_{ij} = 1$, if yes; $w_{ij} = 0$, if no). We then pre-cluster the data to expedite the computational resources required to solve the hard clustering problem by (i) identifying genes with similar experimental responses, and (ii) removing outliers deemed not to be significant to the clustering process. To provide just adequate discriminatory characteristics so that the genes can be pre-clustered properly, we reduce the expression vectors into a set of representative variables $\{+, \text{o}, -\}$. The (+) variable represents an increase in expression level compared to the previous time point, the (−) variable represents a decrease in expression level from the previous time point, and the (o) variable represents an expression level that does not vary significantly (±10% of change across the time points). We could have used other comparative metrics such as distance or correlation to pre-cluster the genes, though at this first pass stage, using the representative variables $\{+, \text{o}, -\}$ lends more ease and produces pre-clusters of similar quality. Obviously the pre-clustering process of choice can differ across datasets to be clustered, and we choose the approach most expeditious to our data of interest.

2.2.2 Hard clustering by global optimization

The global optimization approach seeks to minimize the Euclidean distances between the data points and the centers of their assigned clusters as:

$$\begin{aligned} \underset{w_{ij}, z_{jk}}{\text{Minimize}} \quad & \sum_{i=1}^{n}\sum_{j=1}^{c}\sum_{k=1}^{s} w_{ij}\left(a_{ik} - z_{jk}\right)^2 \qquad \text{(Problem 1)} \\ \text{s.t.} \quad & \sum_{j=1}^{c} w_{ij} = 1, \forall \text{i} = 1, \dots, n \\ & w_{ij} \text{ are binary variables, } z_{jk} \text{ are continuous variables} \end{aligned}$$

There are two sets of variables in the problem, w_{ij} and z_{jk}. While the bounds of w_{ij} are clearly 0 and 1, that of z_{jk} are obtained by observing the range of a_{ik} values.

$$\begin{aligned} z_{jk}^L &= \min\{a_{ik}\}, \forall k = 1, \ldots, s \\ z_{jk}^U &= \max\{a_{ik}\}, \forall k = 1, \ldots, s \end{aligned}$$

The pre-clustering work suggests that some of the genes need only be restricted to some number of known clusters, since it can be determined (for instance by distance and correlation metrics) that certain genes are exceedingly dissimilar from some of the pre-clusters and thus have virtually zero probability of being clustered there. This restriction can be described by introducing an additional binary parameter suit$_{ij}$. A data point deemed to belong uniquely to just one cluster will only have suit$_{ij}$ = 1 for only one value of j and zero for the others, whereas a data point restricted to a few clusters will have suit$_{ij}$ = 1 for only those clusters. This reduces the computational demands of the problem. The introduction of the suit$_{ij}$ parameters also obviates the need for constraints that prevent the redundant re-indexing of clusters. The objective function in Problem 1, when expanded is:

$$\sum_{j=1}^{c} w_{ij} \sum_{i=1}^{n} \sum_{k=1}^{s} a_{ik}^2 - \sum_{i=1}^{n} \sum_{j=1}^{c} \sum_{k=1}^{s} a_{ik} w_{ij} z_{jk} + \sum_{j=1}^{c} \sum_{k=1}^{s} z_{jk} \sum_{i=1}^{n} w_{ij} \left(z_{jk} - a_{ik}\right)$$

Together with the necessary first-order optimality condition:

$$\sum_{i=1}^{n} w_{ij} \left(z_{jk} - a_{ik}\right) = 0, \ \forall j, \forall k$$

(i.e., the vector distance sum of all genes within a cluster to the cluster center should be intuitively zero), and the constraint that each gene is allowed to belong to only one cluster, $\left(\text{i.e. } \sum_{j=1}^{c} w_{ij} = 1\right)$, the formulation becomes:

$$\underset{w_{ij}, z_{jk}}{\text{Minimize}} \ \sum_{i=1}^{n} \sum_{k=1}^{s} a_{ik}^2 - \sum_{i=1}^{n} \sum_{j=1}^{c} \sum_{k=1}^{s} (\text{suit}_{ij})(a_{ik} w_{ij} z_{jk}) \qquad \text{(Problem 2)}$$

$$\begin{aligned} \text{s.t.} \quad & (\text{suit}_{ij}) \left(z_{\text{jk}} \sum_{i=1}^{n} w_{ij} - \sum_{i=1}^{n} a_{ik} w_{ij} \right) = 0, \ \forall j, \forall k \\ & \sum_{j=1}^{c} (\text{suit}_{ij}) w_{ij} = 1, \ \forall i \\ & 1 \le \sum_{j=1}^{n} (\text{suit}_{ij}) w_{ij} \le n - c + 1 \\ & w_{ij} = 0 - 1, \forall i, \forall j \\ & z_{jk}^{\text{L}} \le z_{jk} \le z_{jk}^{\text{U}}, \forall j, \forall k \end{aligned}$$

The first set of constraints are the necessary optimality conditions, the second demand that each gene can belong to only one cluster, and the third state that there is at least one and no more than $(n - c + 1)$ data points in a cluster. Note also that the $\sum_{i=1}^{n} \sum_{k=1}^{s} a_{ik}^2$ term in the objective function of Problem 2 is a constant and can be dropped, though for the sake of completeness we will retain the term throughout the subsequent formulations in the paper. Problems 1 and 2 are Mixed Integer Nonlinear Programming (MINLP) problems with bilinear terms in the objective function and the first set of constraints. To handle the nonlinearities formed by the product of variables w_{ij} and z_{jk}, new variables y_{ijk} along with additional constraints (Floudas 1995) are defined as follows:

$$y_{ijk} = w_{ij} z_{jk} \tag{1}$$

$$z_{jk} - z_{jk}^{U}\left(1 - w_{ij}\right) \leq y_{ijk} \leq z_{jk} - z_{jk}^{L}\left(1 - w_{ij}\right) \tag{2}$$

$$z_{jk}^{\mathrm{L}} w_{ij} \leq y_{ijk} \leq z_{jk}^{U} w_{ij}, \ \forall i, \forall j, \forall k \tag{3}$$

The introduction of y_{ijk} and the additional constraints reduces the formulation to an equivalent Mixed-Integer Linear Programming (MILP) problem, but results in an inordinately large number of variables. Thus, there is a need for new approaches to address large datasets.

2.2.3 The GOS algorithm for clustering

The introduction of the bilinear variable y_{ijk} results in a large number of variables to be considered. In a problem with over 2000 data points, each having 24 features, to be placed into over 380 clusters, the number of variables to be considered numbers over 18 million. Without introducing the y_{ijk} variables will leave the problem in a nonlinear form. Mixed-integer nonlinear programming (MINLP) problems are considered extremely difficult. Theoretical advances and prominent algorithms for solving MINLP problems are addressed in Floudas (1995), Floudas (2000), and Floudas et al. (2005).

The general form of a MINLP problem is:

$$\begin{aligned} Z = \quad & \min_{x,y} C(x,y) & \text{(Problem 3)} \\ \text{s.t.} \quad & h(x,y) = 0 \\ & g(x,y) \leq 0 \\ & y \in (0,1)^m,\ x \in \Re^n \end{aligned}$$

Here, x represents the continuous variables in real space and y, the integer variables. For simplicity here, y is assumed to be binary. In addition, $C(x, y)$ is the objective function, $h(x, y)$ represents the set of equality constraints, and $g(x, y)$ is the set of inequality constraints.

We propose here a variant of the Generalized Benders Decomposition (GBD) algorithm (Geoffrion 1973; Floudas et al. 1989), denoted as the Global Optimum Search (GOS). For brevity, only a outline of the GOS algorithm is presented here, while a more detailed description can be found in Floudas et al. (1989) and Floudas (1995). Successful applications of the decomposition principles of the GOS approach in process synthesis are reported in Floudas and Grossman (1987), Floudas and Anastasiadis (1988), Paules and Floudas (1989), Ciric and Floudas (1989), Aggarwal and Floudas (1990), and Kokossis and Floudas (1994).

In brief, the GBD method decomposes the problem into a primal problem and the master problem. The former optimizes the continuous variables while fixing the integer variables and provides an upper bound solution, while the latter optimizes the integer variables while fixing the continuous variables and provides a lower bound solution. The two sequences of upper and lower bounds are iteratively updated until they converge in a finite number of iterations. In addition, the GOS algorithm assumes that (i) the optimal solution of the primal problem together with the relevant Lagrange multipliers can be used to determine the support functions, (ii) $f(x, y)$ and $g(x, y)$ are convex functions in y for every fixed x, and (iii) $h(x, y)$ are linear functions in y for every x. An outline of the GOS algorithm is as follows:

Step1—Solving the primal problem
The primal problem results from fixing the binary variables to a particular 0–1 combination. Here, w_{ij} is fixed and z_{jk} is solved from the resultant linear programming (LP) problem. In addition, the solution also includes the relevant Lagrange multipliers. The objective function obtained is the upper bound solution. The general form of the feasible problem is:

$$\begin{aligned} Z = \quad & \min_x C(x, y^k) \\ \text{s.t.} \quad & h\left(x, y^k\right) = 0 \\ & g\left(x, y^k\right) \leq 0 \\ & x \in \Re^n \end{aligned} \qquad \text{(Problem 4)}$$

If the primal problem is found to be infeasible, the inactive (i.e., inequality) constraints are relaxed by introducing slack variables α and then solving for α, as well as z_{jk} and the Lagrange multipliers. In this event, no new upper bound solution is found. The infeasible problem to be solved has the form:

$$\begin{aligned} \min \quad & \sum \alpha \\ \text{s.t.} \quad & h\left(x, y^k\right) = 0 \\ & g\left(x, y^k\right) \leq \alpha, \ \text{one } \alpha \text{ value for each inactive constraint} \\ & \alpha \geq 0 \end{aligned} \qquad \text{(Problem 5)}$$

Step 2—Solving the relaxed master problem
The master problem is essentially the problem projected onto the y-space (i.e., that of the binary variables). To expedite the solution of this projection, the dual representation of the master is used. This dual representation is in terms of the supporting Lagrange functions of the projected problem. It is assumed that the optimal solution of the primal problem as well as its Lagrange multipliers can be used for the determination of the support function. Also, the support functions are gradually built up over each successive iteration. The relaxed master problem to be solved is hence:

$$\begin{aligned} \min_{y, \mu_B} \quad & \mu_B \\ \text{s.t.} \quad & \mu_B \geq L(x^k, y, \lambda^k, \mu^k), k = 1, K \\ & 0 \geq \overline{L}(x^l, y, \overline{\lambda}^l, \overline{\mu}^l), l = 1, L \\ & L(x^k, y, \lambda^k, \mu^k) = f(x^k, y) + \lambda^k h(x^k, y) + \mu^k g(x^k, y) \\ & \overline{L}(x^l, y, \overline{\lambda}^l, \overline{\mu}^l) = \overline{\lambda}^l h(x^l, y) + \overline{\mu}^l g(x^l, y) \end{aligned} \qquad \text{(Problem 6)}$$

In accordance to the general format of the problem, $f(x, y)$ is the objective function, $h(x, y)$ are the active constraints, and $g(x, y)$ are the inactive constraints. 'x' represents the solutions of the continuous variables (i.e., z_{jk}) from the primal problem and 'y' represents the binary variables (i.e., w_{ij}) to be determined in the relaxed master. 'λ' represents the Lagrange multipliers for the active constraints and 'μ' represents the Lagrange multipliers for the inactive constraints. The superscript 'k' represents values from the feasible primal problems and the superscript 'l' (and the over-bar) represents values from the infeasible primal problems. The master problem is to be solved as a MIP (mixed integer programming) problem.

The solution loop then returns to Step 1 and the process is repeated. For each excursion into Step 1, the primal could be either feasible or infeasible.

With each successive iteration, a new support function is added to the list of constraints for the master problem. Thus in a sense, the support functions for the master problem build up with each iteration, forming a progressively tighter envelope and gradually pushing up the lower bound solution until it converges with the upper bound solution.

Since fixing x to the solution of the corresponding primal problem may not necessarily produce valid support functions, the master solution obtained at each iteration is checked against the current lower bound solution so that the latter is updated only if the master solution is higher than the current lower bound solution.

With fixed starting values for w_{ij}, the primal problem becomes:

$$\underset{z_{jk}}{\text{Minimize}} \quad \sum_{i=1}^{n}\sum_{k=1}^{s} a_{ik}^2 - \sum_{i=1}^{n}\sum_{j=1}^{c}\sum_{k=1}^{s} a_{ik} w_{ij}^{*} z_{jk} \qquad \text{(Problem 7.1)}$$

$$\text{s.t.} \quad z_{jk}\sum_{i=1}^{n} w_{ij}^{*} - \sum_{i=1}^{n} a_{ik} w_{ij}^{*} = 0,\ \forall j, \forall k$$

$$z_{jk}^{L} \le z_{jk} \le z_{jk}^{U}, \forall j, \forall k$$

The primal problem is a Linear Programming (LP) problem. All the other constraints drop out since they do not involve z_{jk}, which are the variables to be solved in the primal problem. Besides z_{jk}, the Lagrange multipliers λ_{jk}^{m} for each of the constraints above is obtained. The objective function is the upper bound solution. These are inputted into the master problem, which becomes:

$$\min_{w_{ij},\mu_B} \mu_B \qquad \text{(Problem 7.2)}$$

$$\text{such that } \mu_B \ge \sum_{i=1}^{n}\sum_{k=1}^{s} a_{ik}^2 - \sum_{i=1}^{n}\sum_{j=1}^{c}\sum_{k=1}^{s} a_{ik} w_{ij} z_{jk}^{*}$$

$$+\sum_{j=1}^{c}\sum_{\text{k}=1}^{\text{s}} \lambda_{jk}^{m*}\left(z_{jk}^{*}\sum_{i=1}^{n} w_{ij} - \sum_{i=1}^{n} a_{ik} w_{ij}\right), m = 1, M$$

$$\sum_{j} = 1^{c} w_{ij} = 1,\ \forall i$$

$$1 \le \sum_{j=1}^{n} w_{ij} \le n - c + 1, \forall j$$

$$w_{ij} = 0 - 1, \forall i, \forall j$$

The master problem solves for w_{ij} and μ_B, and results in a lower bound solution (i.e., the objective function). The master problem is a Mixed Integer Linear Programming (MILP) problem. The w_{ij} solutions are cycled back into the primal problem and the process is repeated until the solution converges. Thus, there is no longer a need for the variables y_{ijk}, which substantially reduces the number of variables to be solved. Also, after every solution of the master problem, where a solution set for w_{ij} is generated, an integer cut is added for subsequent iterations to prevent redundantly considering that particular solution set again. The cut is expressed as:

$$\sum_{i\in\{n|w_{ij}=1\}}^{n} w_{ij} - \sum_{i\in\{n|w_{ij}=0\}}^{n} w_{ij} \leq n-1 \tag{4}$$

Note that the initial condition w_{ij} for the primal problem can be generated either by solving the above problem as a relaxed MINLP problem or by randomly generating starting w_{ij} values. For the former, the w_{ij} solution is then rounded up and used as the initial condition for the GOS algorithm. It is found that well over 95% of the w_{ij} solution from the MINLP problem adopt [0,1] solutions anyway. In addition, if it is not certain that the initial w_{ij} values form an optimal solution, such as the case of randomly generated w_{ij} values, it is then not included in subsequent integer cuts. It is important to note that while the GOS algorithm tends to give good optimal solutions, it does not have a theoretical guarantee of returning a globally optimum solution. Hence the issue of providing the algorithm with a quality initialization point is important and will be addressed later in the paper. Note also that in a typical GBD algorithm, there may be an infeasible primal problem, for which the problem statement would have to be reformulated accordingly. In this case, since there is only one set of continuous variable (i.e., z_{jk}) to be solved in the primal problem, and there is always a feasible assignment of points to clusters, leading to the calculation of the cluster centers, all primal problems are feasible.

2.2.4 Determining the optimal number of clusters

Most clustering algorithms do not contain screening functions to determine the optimal number of clusters. Yet this is important to evaluate the results of cluster analysis in a quantitative and objective fashion. On the other hand, while it is relatively easy to propose indices of cluster validity, it is difficult to incorporate these measures into clustering algorithms and appoint thresholds on which to define key decision values (Jain and Dubes 1988; Halkidi et al. 2002). Some of the indices used to compute cluster validity include the Dunn's validity index (Dunn 1974), the Davis-Bouldin validity index (Davies and Bouldin 1979), the Silhouette validation technique (Rousseeuw 1987), the C index (Hubert and Schultz 1976), the Goodman–Kruskal index (Goodman and Kruskal 1954), the Isolation index (Pauwels and Frederix 1999), the Jaccard index (Jaccard 1912), and the Rand index (Rand 1971). We note that the optimal number of clusters occurs when the inter-cluster distance is maximized and the intra-cluster distance is minimized. We adapt the concept of a clustering balance (Jung et al. 2003), where it has been shown to have a minimum value when intra-cluster similarity is maximized and inter-cluster similarity is minimized. This provides a measure of how optimal is a certain number of clusters used for a particular clustering algorithm. We introduce the following:

$$\text{Global Center, } z_k^o = \frac{1}{n}\sum_{i=1}^{n} a_{ik},\ \forall k \tag{5}$$

$$\text{Intra-cluster error sum, } \Lambda = \sum_{i=1}^{n}\sum_{j=1}^{c}\sum_{k=1}^{s} w_{ij}\left\| a_{ik} - z_{jk}\right\|_2^2 \tag{6}$$

$$\text{Inter-cluster error sum, } \Gamma = \sum_{j=1}^{c}\sum_{k=1}^{s}\left\| z_{jk} - z_k^o\right\|_2^2 \tag{7}$$

For agglomerative clustering, the intra-cluster error sum increases as the clustering progresses whilst the inter-cluster error sum decreases. For a divisive algorithm, the opposite trend holds. Based on this, Jung et al. (2003) proposed a clustering balance parameter, which is the α-weighted sum of the two error sums.

$$\textit{Clustering}\ \text{Balance, } \varepsilon = \alpha\Lambda + (1-\alpha)\,\Gamma \tag{8}$$

We note here that the rightful α-ratio is 0.5. There are two ways to come to this conclusion. We note that the factor α should balance the contributive weights of the two error sums to the clustering balance. At extreme cluster numbers, that is, the largest and smallest number possible, the sum of the intra-cluster and inter-cluster error sums at both cluster numbers should be balanced. In the minimal case, all the data points can be placed into a single cluster, in the case of which the inter-cluster error sum is zero and the intra-cluster error sum can be calculated with ease. In the maximal case, each data point forms its own cluster, in the case of which the intra-cluster error sum is zero and the inter-cluster error sum can be easily found. Obviously the intra-cluster error sum in the minimal case and inter-cluster error sum in the maximal case are equal, suggesting that the most appropriate weighting factor to use is in fact 0.5. The second approach uses a clustering gain parameter proposed by Jung et al. (2003). This gain parameter is the difference between the decreased inter-cluster error sum γ_j compared to the initial stage and the increased intra-cluster error sum λ_j compared to the initial stage, and is given by.

$$\gamma_{jk} = \sum_{i=1}^{n} w_{ij}\left\| a_{ik} - z_k^o\right\|_2^2 - \left\| z_{jk} - z_k^o\right\|_2^2,\ \forall j, \forall k \tag{9}$$

$$\lambda_{jk} = \sum_{i=1}^{n} w_{ij}\left\| a_{ik} - z_{jk}\right\|_2^2,\ \forall j, \forall k \tag{10}$$

$$\text{Gain, } \Delta_{jk} = \sum_{i=1}^{n} w_{ij}\left\| a_{ik} - z_k^o\right\|_2^2 - \left\| z_{jk} - z_k^o\right\|_2^2 - \sum_{i=1}^{n} w_{ij}\left\| a_{ij} - z_{jk}\right\|_2^2,\ \forall j, \forall k \tag{11}$$

With the identities:

$$\sum_{i=1}^{n} w_{ij}a_{ik} = n_j z_{jk}, \forall j,\ \forall k$$
$$\sum_{i=1}^{n} w_{ij} = n_j, \forall j$$

where n_j denotes the number of data points in cluster j, (11) can be simplified to:

$$\Delta_{jk} = \left(n_j - 1\right) \left\| z_k^o - z_{jk} \right\|_2^2, \forall j, \forall \mathrm{k} \tag{12}$$

$$\therefore \Delta = \sum_{j=1}^{c} \sum_{k=1}^{s} \left(n_j - 1\right) \left\| z_k^o - z_{jk} \right\|_2^2 \tag{13}$$

Jung et al. (2003) showed the clustering gain to have a maximum value at the optimal number of clusters, and demonstrated that the sum total of the clustering gain and balance parameters is a constant. As can be seen from the following derivation, this is only possible if the α-ratio is 0.5.

$$\begin{aligned}
&\text{Sum of Clustering Balance and Clustering Gain,}\\
\Omega &= \varepsilon + \Delta \\
&= \Lambda + \Gamma + \Delta \\
&= \left[\sum_{i=1}^{n}\sum_{j=1}^{c}\sum_{k=1}^{s} w_{ij} \left\| a_{ik} - z_{jk} \right\|_2^2\right] + \left[\sum_{j=1}^{c} \left\| z_{jk} - z_k^o \right\|_2^2\right] + \cdots \\
&\quad \left[\begin{array}{l} \displaystyle\sum_{i=1}^{n}\sum_{j=1}^{c}\sum_{k=1}^{s} w_{ij} \left\| a_{ik} - z_k^o \right\|_2^2 - \\ \displaystyle\sum_{j=1}^{c} \left\| z_{jk} - z_k^o \right\|_2^2 - \sum_{\mathrm{i}=1}^{\mathrm{n}}\sum_{j=1}^{c}\sum_{k=1}^{s} w_{ij} \left\| a_{ik} - z_{jk} \right\|_2^2 \end{array}\right] \\
&= \sum_{\mathrm{i}=1}^{n}\sum_{j=1}^{c}\sum_{k=1}^{s} w_{ij} \left\| a_{ik} - z_k^o \right\|_2^2 \\
&= \sum_{i=1}^{n}\sum_{k=1}^{s} \left\| a_{ik} - z_k^o \right\|_2^2, \text{ which is a constant for any given dataset.}
\end{aligned} \tag{14}$$

These derivations suggest that for any clustering algorithm including that using the GOS algorithm, one can deduce the optimal number of clusters by performing multiple repetitions of the clustering process over a suitably large range of cluster numbers and watching for the clustering gain or clustering balance turning points.

2.3 Proposed algorithm

The discussion thus far points to the GOS formulation as a suitable clustering algorithm. But for it to be effective, the formulation must be provided with a good initialization point. Also, we want to expeditiously incorporate the approach to predict the optimal number of clusters into a clustering algorithm. With these considerations in mind, we propose the following GOS clustering algorithm with enhanced data point positioning (EP_GOS_Clust).

2.3.1 Gene pre-clustering

We choose to pre-cluster genes based on the feature pattern representation of their expression vectors. This conforms well to the intuitive notion that two co-expressed genes similarly shaped expression patterns, rather than comparing the magnitudes of the two series of measurements (Eisen et al. 1998). In our 24-dimensional expression vectors, only genes with two or less different expression vector points from one another are pre-clustered together. Many of these genes end up belonging to more than one pre-cluster. Since their specific membership is in question, we take the clusters formed only by uniquely clustered genes. As a result, we find 388 genes uniquely placed into 157 clusters.

2.3.2 Iterative clustering

We let the initial cluster set be defined by the unique genes pre-clustered in the previous step and compute the cluster centers. We next compute the distance between each of the remaining genes and these initial centers and as a good initialization point placed these genes into the nearest cluster based on:

$$\text{Min}\left\{\sum_{k=1}^{s}\left(a_{ik}-z_{jk}^{\text{initial}}\right)^2, \forall j\right\}, \forall i \notin \text{unique}$$

We then create a rank-order list for each of the remaining genes for its distance to each of the initial clusters, and for each gene allow its suitability in 4 nearest clusters via its suit$_{ij}$ parameters. For this particular dataset, a separate study (results not shown here) has indicated that the clustering results cease to change significantly once the number of suit$_{ij}$ values for each gene exceed 4. The initialization point and the suit$_{ij}$ parameter assignments are then utilized in the primal problem of the GOS algorithm as described in Problem 7.1 to solve for z_{jk}. These, together with the Lagrange multipliers, are inputted into the master problem (Problem 7.2) to solve for w_{ij}. The primal problem gives an upper bound solution and the master problem provides a lower bound. The optimal solution is obtained when the lower and upper bounds converge. Then, the worst placed gene based on:

$$\text{Max}\left\{\sum_{j=1}^{c}\sum_{k=1}^{s}\left(a_{ik}-z_{jk}^{\text{updated}}\right)^2, \forall i \notin \text{unique}\right\}$$

is removed and used as a seed for a new cluster with center z^{new}. This gene has already been subjected to the initial search for membership so there is no reason for it to belong to any one of the older clusters. Based on z^{new} and z^{updated} (updated without the worst-placed gene), the iterative steps are repeated, by selecting a new initialization point, assignment suit$_{ij}$ parameters, and running the GOS algorithm again. With these iterations, the number of clusters builds up from the initial number defined by the pre-clustering work, until the optimal number of clusters is attained. Our proposed clustering methodology can be summarized by the schematic in Fig. 1.

3 Results and discussion

3.1 Description of comparative study

We will work with the 5,652 genes obtained previously. The clustering algorithms to be compared are (a) K-Means, (b) K-Medians, (c) K-Corr, where the Pearson correlation coefficient is the distance metric, (d) K-CityBlock, where the distance metric is the city block distance, or the 'Manhattan' metric, which is akin to the north-south or east-west walking distance in a place like New York's Manhattan district,

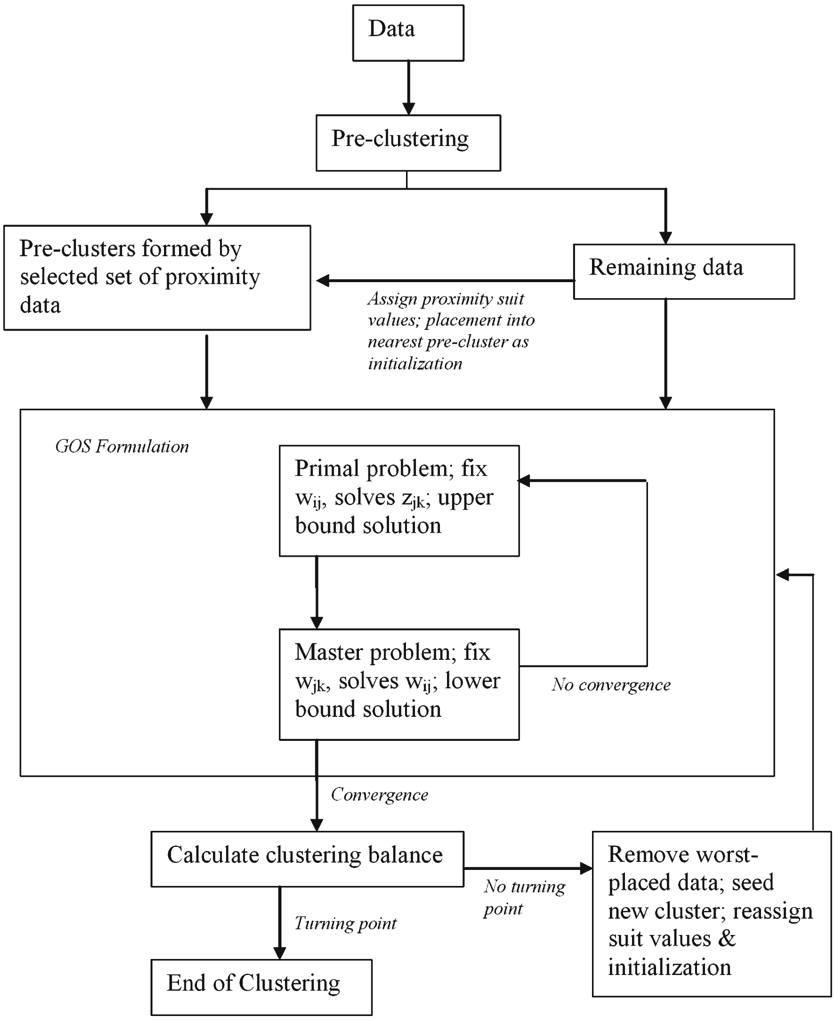

Data
Pre-clustering
Pre-clusters formed by selected set of proximity data
Remaining data
Assign proximity suit values; placement into nearest pre-cluster as initialization
GOS Formulation
Primal problem; fix w_ij, solves z_jk; upper bound solution
Master problem; fix w_jk, solves w_ij; lower bound solution
No convergence
Convergence
Calculate clustering balance
No turning point
Remove worst-placed data; seed new cluster; reassign suit values & initialization
Turning point
End of Clustering

Fig. 1 Schematic flowchart of the EP_GOS_Clust algorithm. Although the formulation in the paper has been notated for DNA microarray data, the algorithm framework can be adapted for clustering any numeric data

(e) K-AvePair, where the cluster metric is the average pair-wise distance between members in each cluster, (f) QTClustering, (g) SOM, (h) SOTA, (i) GOS I, where genes with up to 7 different feature points are pre-clustered, initial clusters are defined by uniquely placed genes, and each gene is placed into its nearest cluster as the initialization point, and (j) EP_GOS_Clust, for which genes are pre-clustered if they have 2 or less different feature points and can be uniquely clustered. For convenience, the comparison involving SOM and SOTA will only be carried out at the optimal cluster number predicted for the EP_GOS_Clust. Since the K-family of clustering approaches are sensitive to the initialization point, we run each 25 times and use only the best result.

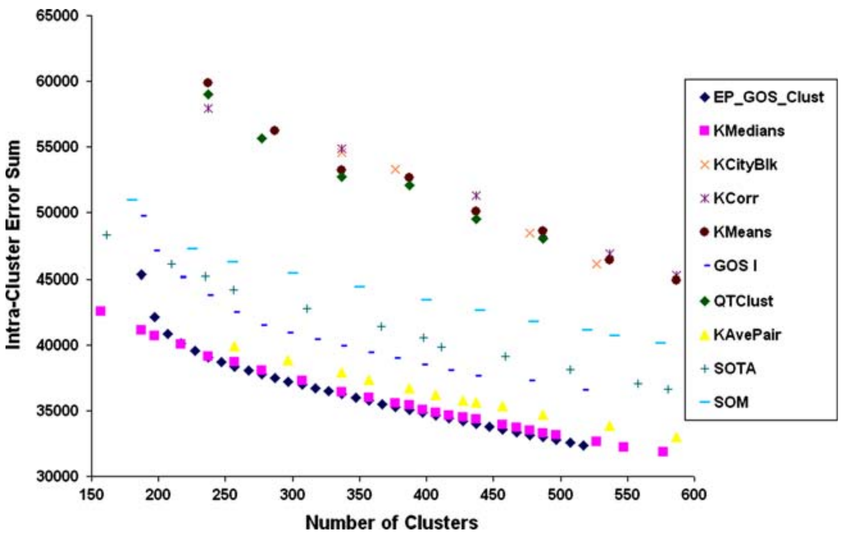


Fig. 2 Comparison of intra-cluster error sum from the clustering of 5,652 Yeast genes based on DNA expression levels in glucose pathway experiments, using different clustering algorithms. Each gene contains 36 time points, or a 24-dimensional feature vector. The intra-cluster error sum measures the extent of dissimilarity between objects within the same cluster, and should be minimized

3.2 Intra-cluster error sum

Data points in the same cluster should be as similar as possible; hence the intra-cluster error sum should be minimized. From Fig. 2, it can be seen that the best performing clustering algorithms are the K-Medians and the EP_GOS_Clust. In fact, other than within regions of low cluster number, the EP_GOS_Clust outperforms all the other algorithms. One reason for the efficacy of the K-Medians at low cluster numbers is due to it using the data median to compute cluster centers. This circumvents the distorting effects of outlier data points, which particularly affects algorithms that use random initialization points, such as K-Means. It is also notable that the GOS I performs admirably even though the pre-clustering allows genes with up to 30% difference in feature points to be grouped together. This reflects the rigor of the subsequent steps in assigning suit$_{ij}$ parameters, the GOS clustering, and the process of incrementing the cluster number. The clustering results also show up the inadequacy of QTClust. It groups genes together till the cluster reaches a pre-determined tolerance. The algorithm then determines the number of clusters to use. A different tolerance criterion needs to be specified in order to obtain a different cluster number. This implies that the process of probing for the optimal number of clusters using QTClust uses clusters of inconsistent qualities. We further look in detail at the clustering results obtained by QTClust and note that genes with up to 14 different feature points (∼60% of all feature points) are in fact clustered together.

3.3 Inter-cluster error sum

This error sum indicates how different clusters are from one another and is given by:

$$\sum_{j=1}^{c}\sum_{k=1}^{s}\left(z_{jk}-z_{k}^{o}\right)^{2}$$

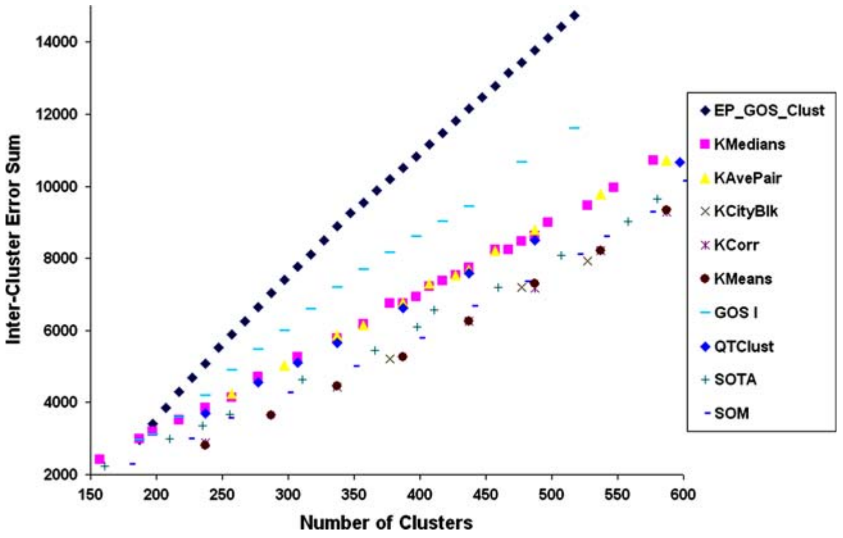


Fig. 3 Comparison of inter-cluster error sum from the clustering of 5,652 yeast genes based on DNA expression levels in glucose pathway experiments, using different clustering algorithms. Each gene contains 36 time points, or a 24-dimensional feature vector. The inter-cluster error sum measure the extent of dissimilarity between different clusters and should be maximized

This is another measure of cluster quality, and it is desirable for the error sum to be maximized. The inter-cluster error sum for the clustering of 5,652 genes is shown in Fig. 3. Here, the EP_GOS_Clust outperforms all the other cluster algorithms. In using the intra-cluster error sum as the objective function and demanding that the worst-fitting gene be extracted to seed new clusters, the EP_GOS_Clust explicitly seeks a minimal intra-cluster error sum and implicitly searches for a configuration that maximizes the inter-cluster error sum. Note that while K-Medians does well in obtaining a minimal intra-cluster error sum, it performs averagely in discerning dissimilar clusters. This is due to the ‘localized’ nature of the K-family of clustering methods, where there is a tendency to become ‘stuck’ within a limited vicinity of the initialization point for most data structures.

3.4 Difference between intra-cluster and inter-cluster error sums

We look also at an overall measure of clustering quality the difference between the intra-cluster and inter-cluster error sums. Since it is desirable for the former to be minimized and the latter maximized, an effective and rigorous clustering algorithm will have a low value for this difference. The results are shown in Fig. 4. Again, the EP_GOS_Clust is the best performer except for certain regions of low cluster number, where the K-Medians dominate with its capability to handle outlier data points. At higher cluster numbers however, the EP_GOS_Clust identifies and isolates these outlier data points into new clusters and subsequently its clustering performance overtakes that of the K-Medians.

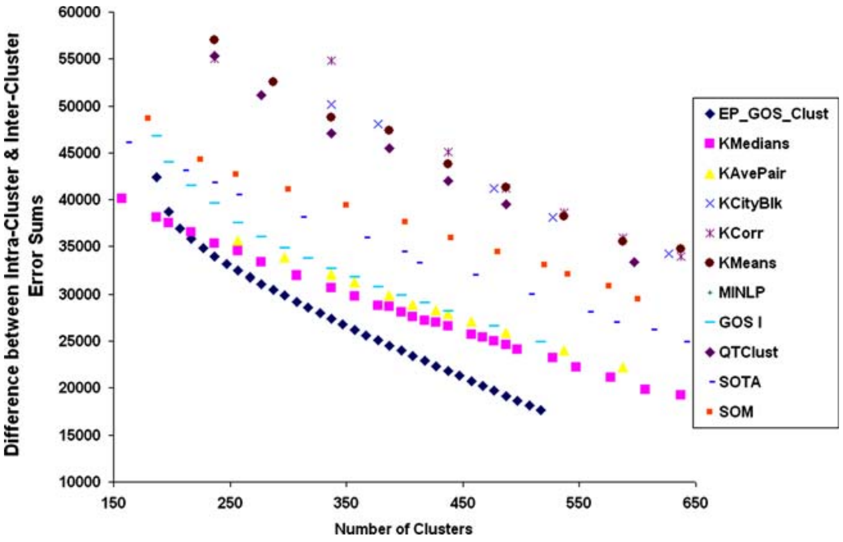


Fig. 4 Comparison of the difference between error sums from the clustering of 5,652 yeast genes based on DNA expression levels in glucose pathway experiments, using different clustering algorithms. Each gene contains 36 time points, or a 24-dimensional feature vector. This comparison allows an overview of the extent of overall 'error-ness' for the clusters formed and should be minimized

3.5 Optimal number of clusters

We compute the optimal number of clusters by applying a suitable weighting factor to the two error sums and then finding the clustering balance. The EP_GOS_Clust predicts the lowest number of optimal clusters. From Fig. 5, it can be seen that EP_GOS_Clust predicts 237 clusters. On the other hand, K-Means and K-Corr, for instance, predict the optimal number of clusters to be around 700, while K-Medians puts the number at around 450. Together with the quality of the EP_GOS_Clust from the previous comparisons, we infer the superior 'economy' of the EP_GOS_Clust in producing tighter data groupings by utilizing a lower number of clusters, as it is actually possible to achieve tight groupings by using a large number of clusters, even with an inferior clustering algorithm.

3.6 Coherence and biological relevance

Often the most intuitive and convenient manner of evaluating the robustness of a clustering approach is to visually inspect the cluster tightness. For brevity, Fig. 6 depicts the expression time course for 6 sample clusters. We use the largest clusters formed, as well as clusters formed in the later stages of the procedure and smaller-sized clusters to show good consistency and lack of size-bias. The plots clearly show the tightness of the clustering throughout. To more conveniently demonstrate the overall tightness of the clusters uncovered by the EP_GOS_Clust as compared to other methods, we also find the Pearson correlation coefficients of all the clusters uncovered. Table 1 provides a summary of the cluster correlations, where in particularly the average correlation coefficient and the standard coefficient reflect the overall tightness for all clusters. It can be seen that the EP_GOS_Clust compares very well with other

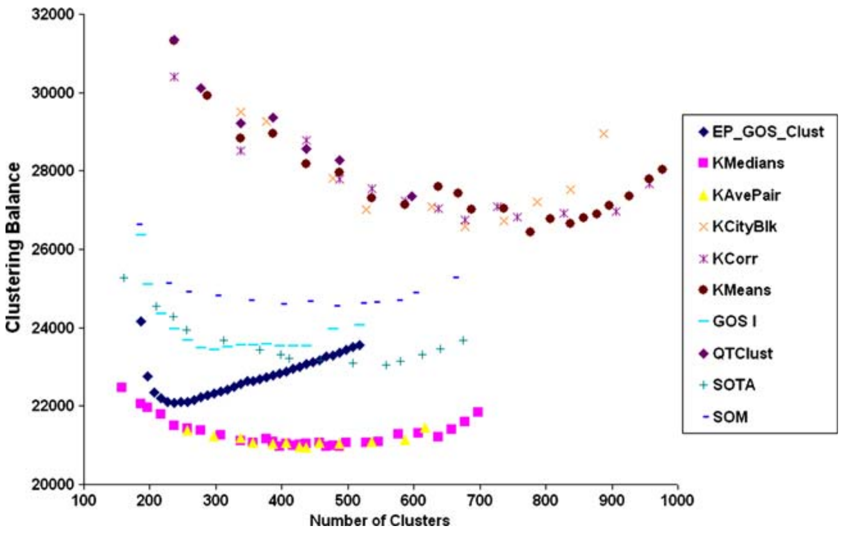


Fig. 5 Prediction of the optimal number of clusters, as shown by the turning point in the cluster balance; by different clustering algorithms from the clustering of 5,652 yeast genes based on DNA expression levels in glucose pathway experiments. Each gene contains 36 time points, or a 24-dimensional feature vector

Table 1 Comparison of cluster correlation from the clustering of 5652 yeast genes based on DNA expression levels in glucose pathway experiments

	Optimal cluster number	Correlation coefficient			
		Average	Maximum	Minimum	Standard deviation
(Clustering Method)					
EP_GOS_Clust	237	0.617*	0.938*	0.264*	0.128*
KMedians	445	0.615	0.937	0.197	0.134
KCityBlk	665	0.398	0.760	−0.159	0.149
KCorr	665	0.630*	0.931	0.239*	0.119*
KMeans	775	0.614	0.959*	0.072	0.131
GOS I	295	0.590	0.933	0.202	0.148
KAvePair	452	0.567	0.909	0.156	0.141
SOTA	540	0.604	0.925	0.378*	0.122*
SOM	485	0.623*	0.968*	0.202	0.156

The comparison shows the average correlation coefficients across all clusters for each clustering algorithm, the maximum and minimum coefficient, as well as the standard deviation of the coefficients to give a sense of the spread of correlation displayed by the clusters. The table also shows the optimal number of clusters predicted by each clustering approach. The shaded row contains the results for EP_GOS_Clust and the top three performers for each correlation performance indicator is marked with an asterisk

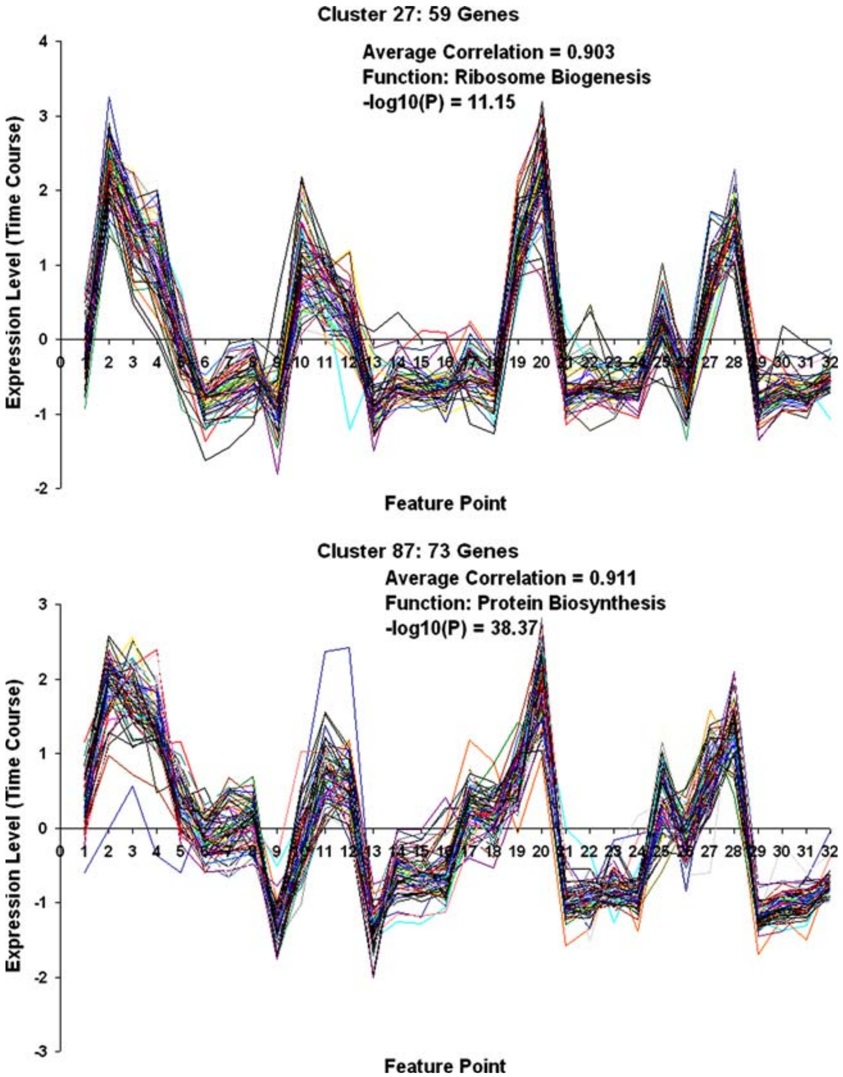
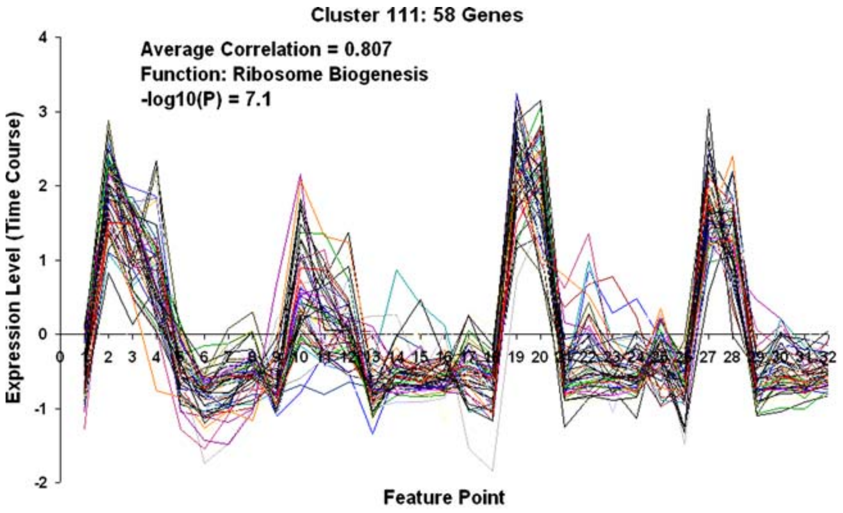


Fig. 6 Gene expression time course plots for 6 sample clusters, found using the EP_GOS_Clust algorithm

clustering methods in producing highly correlated clusters, even against methods such as K-Corr that already explicitly uses correlation as a metric for clustering and the correlation hunting SOM (see Table 1). The data in the table for each clustering algorithm is obtained at the respective optimal number of clusters predicted by the clustering balance.

We also evaluate our clusters by performing a functional search using the Gene Ontology (GO) term finder on the SGD website (http://www.yeastgenome.org). The biological coherence of each cluster is scored according to the percentage of its genes covered by annotations significantly enriched in the cluster. From our results, 91% of the genes group into clusters with p-values under 0.01 and 87% of the genes fall into clusters with p-values under 0.005, which is a significant indication of clustering



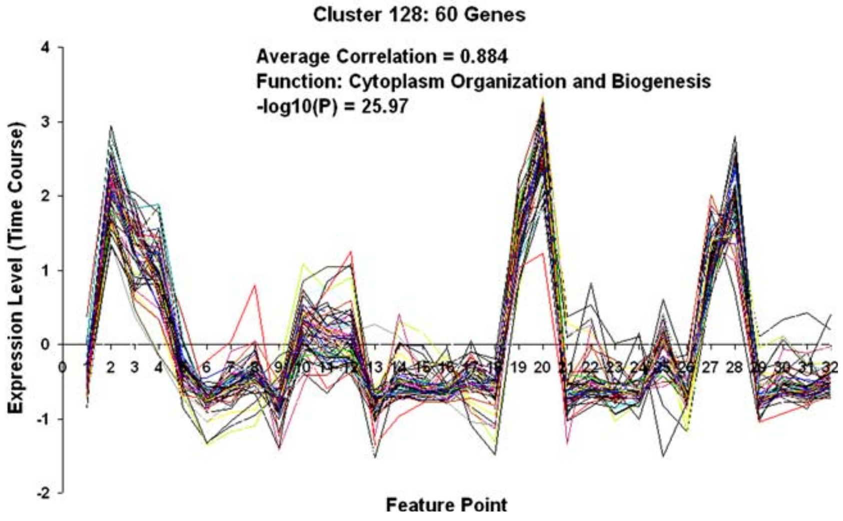
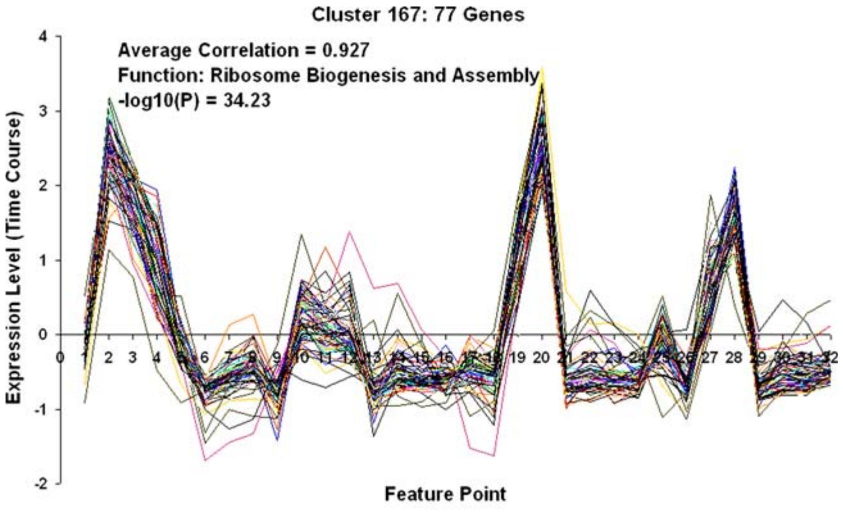


Fig. 6 continued

quality. Table 2 shows that the EP_GOS_Clust performs well against other clustering algorithm in obtaining clusters with good overall *p*-values (expressed as $-\log_{10}(\text{p})$ values in this table) and the proportion of genes that are placed into significantly coherent clusters, which we consider to be two broad tenets in assessing the strength of biological coherence. We would like to point out that the EP_GOS_Clust procedure isolates errant data points as the clustering progresses. Thus, in further analysis of the clusters we have good justification to consider these data points as being irrelevant.

3.7 Additional constraints for large datasets

It is interesting to note that a close examination of the clustering results within each GOS iteration reveals that the cluster size distribution does not change significantly over successive iterations. This suggests that we can analyze the intermediate results



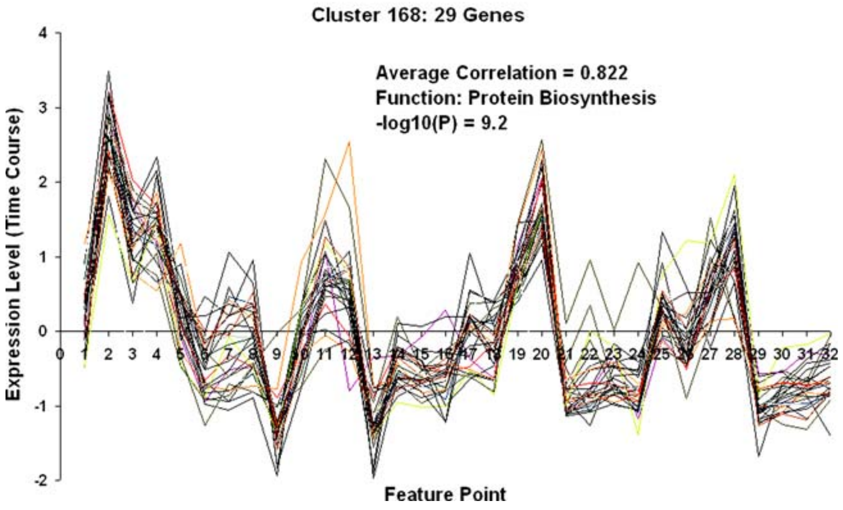


Fig. 6 continued

from a particular run and introduce additional constraints on the number of clusters allowable in each size class without significantly compromising on the optimality of the final solution. Using $1 \leq \sum_{j=1}^{n} w_{ij} \leq n - c + 1$ as the constraint for cluster size can unnecessarily increase the problem size. Hence, we can further tighten the constraint for cluster size as a plausible strategy for further expediting the clustering of even larger data sets.

Indexing the new cluster size ranges by l, for $l = 1, \ldots, d$, we introduce a new binary variable w'_{jl}, which equals one if cluster j belongs to size class l, and zero if otherwise. The additional constraints are then formulated as follows:

$$\sum_{l=1}^{d} w'_{jl} = 1, \ \forall j \tag{15}$$

Table 2 Gene Ontology comparison between Clusters found by different clustering approaches

	$-\log_{10}(P)$ Comparison		%Genes (Total 5652)	
	Average deviation	Standard	In clusters with $-\log_{10}(P)$ values $>=4$	In clusters with $-\log_{10}(P)$ values $>=3$
(Clustering Method)				
EP_GOS_Clust	4.40*	0.37	32.82*	64.92*
KMedians	4.27*	0.34*	30.83*	62.23*
KCityBlk	3.69	0.49	27.53	56.68
KCorr	4.15*	0.39	32.59*	60.08*
KMeans	3.45	0.41	25.11	55.20
GOS I	3.84	0.42	28.19	57.75
KAvePair	3.77	0.48	25.18	54.43
SOTA	3.67	0.31*	30.20	58.86
SOM	3.94	0.35*	30.47	59.24

The table compares the $-\log_{10}(P)$ values of the clusters, which reflect the level of annotative richness, as well as the proportion of yeast genes that fall into biologically significant clusters. The latter is important in 'presenting' the maximal amount of relevant genetic information for follow-up work in areas such as motif recognition and regulatory network inference. The shaded row contains the results for EP_GOS_Clust and the top three performers for each performance indicator is marked with an asterisk

$$n_c - \varepsilon \le \sum_{j=1}^{c} w'_{jl} \le n_c + \varepsilon, \ \forall l \tag{16}$$

$$\sum_{l=1}^{d} d_{l,\min} w'_{jl} \le \sum_{i=1}^{n} w_{ij} \le \sum_{l=1}^{d} d_{l,\max} w'_{jl}, \ \forall j \tag{17}$$

The first set of constraints allows each cluster into only one size class. The second constraint restricts the number of clusters allowable in each class. The parameter ε is judiciously picked to allow a reasonable range over the number of clusters in each class, for instance 10%. Finally, the third constraint bounds the size of the clusters allowed in each class. These additional constraints also involve the variable w_{ij} but not z_{jk}; hence they are all included into the master problem.

4 Conclusion

In our study, we propose a novel clustering algorithm (EP_GOS_Clust) based on a Mixed-Integer Nonlinear Programming (MINLP) formulation. We test our proposed algorithm on a substantially large dataset of gene expression patterns from the yeast *Saccharomyces cerevisiae*, and show that our method compares favorably (if not outperforms) with other clustering methods in identifying data points that are the most similar to one another as well as identifying clusters that are the most dissimilar to one another. We also show that the EP_GOS_Clust is capable of uncovering tightly correlated clusters. Given the nature of the test datasets, we too show that the EP_GOS_Clust does well in uncovering clusters with good biological coherence. In addition, we demonstrate the utility of the pre-clustering procedure and a methodology that works in concert with the algorithm itself to predict the optimal number of clusters. For consistency, we repeated our study on other DNA microarray datasets based on the various glucose signaling pathways in the yeast *Saccharomyces cerevisiae* (other results not reported here) and obtained similar result trends.

Acknowledgements The authors gratefully acknowledge support from the National Science Foundation and the National Institutes of Health.

References

Adams, W.P., Sherali, H.D.: Linearization strategies for a class of zero-one mixed integer programming problems. Operat. Res. **38**(2), 217–226 (1990)

Aggarwal, A., Floudas, C.A.: Synthesis of general separation sequences - nonsharp separations. Comput. Chem. Eng. **14**, 631–653 (1990)

Beer, M., Tavazoie, S.: Predicting gene expression from sequence. Cell **117**, 185–198 (2004)

Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York (1981)

Brooke, A., Kendrick, D., Meeraus, A.: GAMS: A User's Guide. The Scientific Press, San Francisco, CA (1988)

Carpenter, G., Grossberg, S.: ART3: hierarchical search using chemical transmitters in self-organizing patterns recognition architectures. Neural Networks **3**, 129–152 (1990)

Ciric, A.R., Floudas, C.A.: A retrofit approach of heat exchanger networks. Comput. Chem. Eng. **13**, 703-715 (1989)

Claverie, J.: Computational methods for the identification of differential and coordinated gene expression. Human Mol. Genet. **8**, 1821–1832 (1999)

Davis, D.L., Bouldin, D.W.: A cluster separation measure. IEEE Trans. Pattern Anal. Machine Intell. **1**(4), 224–227 (1979)

Dempster, A.P., Laird, N.M., Rudin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. Roy. Stat. Soc. B. **39**(1), 1–38 (1977)

DeRisi, J.L., Iyer, V.R., Brown, P.O.: Exploring the metabolic and genetic control of gene expression on a genomic scale. Science **278**, 680–686 (1997)

Dhillon, I.S., Guan, Y.: Information theoretic clustering of sparse co-occurrence data. Proceedings of the Third IEEE International Conference on Data Mining (ICDM) (2003)

Dunn, J.C.: A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. J. Cybernet. **3**, 32–57 (1973)

Dunn, J.C.: Well separated clusters and optimal fuzzy partitions. J. Cybernet. **4**, 95–104 (1974)

Duran, M.A., Odell, P.L.: Cluster Analysis: A Survey. Springer Verlag, New York (1974)

Eisen, M.B., Spellman, P.T., Brown, P.O., Botstein, D.: Cluster analysis and display of genome-wide expression patterns. Proc. Nat. Acad. Sci. U.S.A. **95**(25), 14863–14868 (1998)

Floudas, C.A., Akrotirianakis, I.G., Caratzoulas, S., Meyer, C.A., Kallrath, J.: Global optimization in the 21st Century: advances and challenges. Comput. Chem. Eng. **29**, 1185–2002 (2005)

Floudas, C.A. Deterministic Global Optimization: Theory, Algorithms, and Applications. Kluwer Academic Publishers (2000)

Floudas, C.A.: Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications. Oxford University Press (1995)

Floudas, C.A., Aggarwal, A., Ciric, A.R.: Global optimum search for non convex NLP and MINLP problems. Comp. Chem. Eng. **13**(10), 1117–1132 (1989)

Floudas, C.A., Anastasiadis, S.H. Synthesis of general distillation sequences with several multicomponent feeds and products. Chem. Eng. Sci. **43**, 2407-2419 (1988)

Floudas, C.A., Grossmann, I.E.: Synthesis of flexible heat exchanger networks with uncertain flow rates and temperatures. Comput. Chem. Eng. **11**, 319-336 (1987)

Geoffrion, A.M.: Generalized benders decomposition. J. Optim. Theory Appl. **10**(4), 237 (1973)

Goodman, L., Kruskal, W.: Measures of associations for cross-validations. J. Am. Stat. Assoc. **49**, 732–764 (1954)

Gower, J.C., Ross, G.J.S.: Minimum spanning trees and single-linkage cluster analysis. Appl. Stat. **18**, 54–64 (1969)

Halkidi, M., Batistakis, Y., Vazirgiannis, M.: Cluster validity methods: Part 1. SIGMOD Record **31**(2), 40–45 (2002)

Hansen, P., Jaumard, B.: Cluster analysis and mathematical programming. Math. Program. **79**, 191–215 (1997)

Hartigan, J.A.: Clustering Algorithms. John Wiley & Sons, New York (1975)
Hartigan, J.A., Wong, M.A.: Algorithm AS 136: a K-means clustering algorithm. Appl. Stat. J. Roy. St. C. **28**, 100–108 (1979)
Herrero, J., Valencia, A., Dopazo, J.: A hierarchical unsupervised growing neural network for clustering gene expression patterns. Bioinformatics **17**(2), 126–136 (2001)
Heyer, L.J., Kruglyak, S., Yooseph, S.: Exploring expression data: identification and analysis of co-expressed genes. Genome Res. **9**, 1106–1115 (1999)
Hubert, L., Schultz, J.: Quadratic assignment as a general data-analysis strategy. Br. J. Math. Stat. Psychol. **29**, 190–241 (1976)
Jaccard, P.: The distribution of flora in the alpine zone. New Phytol. **11**, 37–50 (1912)
Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Comput. Surv. **31**(3), 264–323 (1999)
Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice-Hall Advanced Reference Series, Prentice-Hall, Inc., Englewood Cliffs, New Jersey (1988)
Johnson, R.E.: The role of cluster analysis in assessing comparability under the US transfer pricing regulations. Business Economics (April 2001)
Jung, Y., Park, H., Du, D., Drake, B.L.: A decision criterion for the optimal number of clusters in hierarchical clustering. J. Global Optimiz. **25**, 91–111 (2003)
Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science **220**(4598), 671–680 (1983)
Kohonen, T.: Self Organization and Associative Memory. Springer Information Science Series, Springer Verlag, Berlin, Heidelberg, New York (1984)
Kohonen, T.: Self-Organizing Maps. Springer Verlag, Berlin (1997)
Kokossis, A.C., Floudas, C.A.: Optimization of complex reactor networks - II. Nonisothermal operation. Chem. Eng. Sci. **49**, 1037-1051 (1994)
Leisch, F., Weingessel, A., Dimitriadou, E.: Competitive learning for binary valued data. In: Niklasson L., Bod'en M., Ziemke T. (eds.) Proceedings of the 8th International Conference on Artificial Neural Networks (ICANN 98), vol. 2, pp. 779–784. Sk"ovde, Sweden, Springer (1998)
Likas, A., Vlassis, N., Vebeek, J.L.: The global K-means clustering algorithm. Pattern Recogn. **36**, 451–461 (2003)
Lin, X., Floudas, C., Wang, Y., Broach, J.R.: Theoretical and computational studies of the glucose signaling pathways in yeast using global gene expression data. Biotechnol. Bioeng. **84**(7), 864–886 (2003)
Lukashin, A.V., Fuchs, R.: Analysis of temporal gene expression profiles: clustering by simulated annealing and determining the optimal number of clusters. Bioinformatics **17**(5), 405–414 (2001)
McQueen, J.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)
Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., Teller, E.J.: Equations of State calculations by fast computing machines. J. Chem. Phys. **21**, 1087–1091 (1953)
Paules, G.E. IV., Floudas, C.A.: APROS: Algorithmic development methodology for discrete-continuous optimization problems. Oper. Res. J. **37**, 902-915 (1989)
Pauwels, E.J., Frederix, G.: Finding salient regions in images: non-parametric clustering for image segmentation and grouping. Comput. Vision Image Understand. **75**, 73–85 (1999)
Pipenbacher, P., Schliep, A., Schneckener, S., Schonhuth, A., Schomburg, D., Schrader, R.: ProClust: improved clustering of protein sequences with an extended graph-based approach. Bioinformatics **18**(Suppl 2), S182–S191 (2002)
Rand, W.M.: Objective criteria for the evaluation of clustering methods. J. Am. Stat. Assoc. **66**(336), 846–850 (1971)
Rousseeuw, P. J.: Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. J. Comp. App. Math. **20**, 53–65 (1987)
Ruspini, E.H.: A new approach to clustering. Inf. Control **15**, 22–32 (1969)
Schneper, L., Düvel, K., Broach, J.R.: Sense and sensibility: nutritional response and signal integration in yeast. Curr. Opin. Microbiol. **7**(6), 624–630 (2004)
Sherali, H.D., Desai, J.: A global optimization RLT-based approach for solving the hard clustering problem. J. Global Optimiz. **32**(2), 281–306 (2005a)
Sherali, H.D., Desai, J.: A global optimization RLT-based approach for solving the fuzzy clustering approach. J. Global Optimiz. **33**(4), 597–615 (2005b)
Slonim, N., Atwal, G.S., Tkačik, G., Bialek, W.: Information based clustering. Proc. Nat. Acad. Sci. U.S.A. **102**(51), 18297–18302 (2005)

Sokal, R.R., Michener, C.D.: A statistical method for evaluating systematic relationships. Univ. Kans. Sci. Bull. **38**, 1409–1438 (1958)

Sorlie, T., Tibshirani, R., Parker, J., Hastie, T., Marron, J.S., Nobel, A., Deng, S., Johnsen, H., Pesich, R., Geisler, S., Demeter, J., Perou, C.M., Lonning, P.E., Brown, P.O., Borresen-Dala, A.L., Botstein, D.: Repeated observations of breast tumor subtypes in independent gene expression data sets. Proc. Nat. Acad. Sci. U.S.A. **100**, 8418–8423 (2003)

Tishby, N., Pereira, F., Bialek, W.: The information bottleneck method; proceedings of the 37th annual allerton conference on communication. Control Comput. 368–377 (1999)

Troyanskaya, O.G., Dolinski, K., Owen, A.B., Altman, R.B., Botstein, D.: A Bayesian framework for combining heterogeneous data sources for gene function prediction (in *Saccharomyces cerevisiae*). Proc. Nat. Acad. Sci. U.S.A. **100**, 8348–8353 (2003)

Wang, Y., Pierce, M., Schneper, L., Guldal, C.G., Zhang, X., Tavazoie, S., Broach, J.R.: Ras and Gpa2 mediate one branch of a redundant glucose signaling pathway in yeast. Plos Biol. **2**(5), 610–622 (2004)

Weiler, J., Gausepohl, H., Hauser, N., Jensen, O.N., Hoheisel, J.D.: Hybridization-based DNA screening on peptide nucleic acid (PNA) oligomer arrays. Nuclei Acids Res. **25**, 2792–2799 (1997)

Wu, Z., Leahy, R.: An optimal graph theoretic approach to data clustering: theory and its application to image segmentation. IEEE Trans. Pattern Recogn. Mach. Intell. **15**(11), 1101–1113 (1993)

Xu, R., Wunsch, D. II.: Survey of clustering algorithms. IEEE Trans. Neural Networks **16**(3), 645–678 (2005)

Zahn, C.T.: Graph theoretical methods for detecting and describing gestalt systems. IEEE Trans. Comput. **C**-**20**, 68–86 (1971)

Zhang, B., Hsu, M., Dayal, U.: K-Harmonic Means – A Data Clustering Algorithm. Hewlett-Packard Research Laboratory Technical Report (June 1999)

Zhang, B.: Generalized K-Harmonic Means: Boosting in Unsupervised Learning. Hewlett-Packard Research Laboratory Technical Report (October 2000)

Computational resources

All optimization formulations are written in GAMS (General Algebraic Modeling System) (Brooke et al. 1988) and solved using the commercial solver CPLEX 8.0. GAMS is a high level modeling system specifically designed for mathematical optimization. It consists of a language compiler and an integrated high performance solver such as CPLEX, DICOPT, or XPRESS.