TECHNIQUES FOR CLUSTERING GENE EXPRESSION DATA

G. Kerr [*], H.J. Ruskin, M. Crane, P. Doolan

Biocomputation Research Lab, (Modelling and Scientific Computing Group, School of Computing) and National Institute of Cellular Biotechnology, Dublin City University, Dublin 9, Ireland.

Abstract

Many clustering techniques have been proposed for the analysis of gene expression data obtained from microarray experiments. However, choice of suitable method(s) for a given experimental dataset is not straightforward. Common approaches do not translate well and fail to take account of the data profile. This review paper surveys state of the art applications which recognises these limitations and implements procedures to overcome them. It provides a framework for the evaluation of clustering in gene expression analyses. The nature of microarray data is discussed briefly. Selected examples are presented for the clustering methods considered.

Key words: Gene Expression, Clustering, Bi-clustering, Microarray Analysis

1 Introduction

Gene expression (GE) is the fundamental link between genotype and phenotype in a species, with microarray technologies facilitating the measurement of thousands of GE values under tightly controlled conditions, e.g. (i) from a particular point in the cell cycle, (ii) after an interval response to some environmental change, (iii) from RNA, isolated from a tissue exhibiting certain phenotypic characteristics and so on. A problem inherent in the use of microarray technologies is the huge amount of data produced. Searching for meaningful information patterns and dependencies among genes, in order to provide a basis for hypothesis testing, typically includes the initial step of

[*] Biocomputation Research Lab, Modelling and Scientific Group, Dublin City University, Dublin 9, Ireland
Email address: gkerr@computing.dcu.ie (G. Kerr).

grouping genes, with similar changes in expression into groups or "clusters". Cluster analysis assumes that there is an unknown mapping that assigns a label to each gene and the goal of cluster analysis is to estimate this mapping, i.e. to assign each gene to a group. It is an exploratory technique commonly applied to GE data, where gene labels are not known. However, common approaches do not always translate well to the analysis of microarray data, and fail in significant ways to account for data profile.

Many excellent reviews of microarray data analysis using clustering techniques are available. Asyali et al. [1] provides a synopsis of class prediction (supervised pattern recognition classification) and class discovery (supervised pattern recognition clustering), while Pham et al. [2] provide a comprehensive literature review at the various stages of data analysis during the microarray experiment. A landmark paper Jain et al [3] provides a thorough introduction to clustering, and gives a taxonomy of clustering algorithms (used in this review). Reviewing the state of the art in GE analysis is complicated by the vast interest in this field, and hence many techniques are available. This review aims to evaluate current techniques, which implemented modifications to address shortcomings of conventional approaches and the properties of GE data.

Multiple expression measurements are commonly recorded as a real-valued matrix, with row objects corresponding to GE measurements over a number of experiments and columns corresponding to the pattern of expression of all genes for a given microarray experiment. Each entry, x_{ij}, is the measured expression of gene i in experiment j. *Dimensionality* of a gene refers to the number of expression values recorded for it (number of matrix columns). A *gene/gene profile* x is a single data item (row) consisting of d measurements, $x = (x_1, x_2, ..., x_d)$. A *experiment/sample* y is a single microarray experiment corresponding to a single column in the GE matrix, $y = (x_1, x_2, ..., x_n)^T$ where n is the number of genes in the dataset.

There are a number of common characteristics of all GE datasets. The *accuracy* of the data strongly depends on experimental design and minimisation of technical variation, whether due to instruments, observer or pre-processing [4]. Image corruption or/and slide impurities may led to *incomplete* data [5]. Generally, the number of missing values increases with the number of genes [6]. Many clustering algorithms require a complete matrix of input values, so imputation or missing data estimation techniques need to be considered before clustering. Microarray data is intrinsically *noisy*, resulting in outliers, typically managed by alleviation by robust statistical estimation/testing, (when extreme values are not of primary interest), or identification (when outlier information of intrinsic importance e.g. caused by known bias or experimental error etc.) [7]. As cluster analysis is usually exploratory, lack of *a priori* knowledge on gene groups or their number, is common. Fixing this number may undesirably bias the search, as pattern elements may be ill-defined unless signals are strong. Meta-data can guide the choice of the correct number of clusters, *e.g. genes with common promoter sequence are likely to be expressed together and thus are likely to be placed in the same group.* [8] and [9] discuss methods for determining optimal number of groups.

Clustering the microarray matrix can be achieved in two ways: (i) genes can form a group which show similar expression across conditions, (ii) samples can form a group which show similar GE across all genes. This gives rise to *global clusterings*, where a gene or sample is *grouped across all dimensions*. However, the interplay of genes and samples can be used to give rise to what is termed *bi-clusters*, whereby genes and samples are clustered simultaneously. Such bi-clusters are defined over a subset of genes and a subset of samples thus capturing *local structures* in the dataset. This is a major strength of bi-clustering as cellular processes are understood to rely on subsets of genes, which are co-regulated and co-expressed under certain conditions and behave independently under others, [10]. Justifiably, this approach has been gaining much interest of late. For an excellent review on bi-clusters and bi-clustering techniques see [11].

Additionally, the clustering can be *complete* or *partial*. A complete clustering assigns each gene to a cluster, whereas a partial clustering does not. Partial clusterings tend to be more suited to GE, as the dataset often contains irrelevant genes or samples. Clearly this allows: (i) "noisy genes" to be left out, with correspondingly less impact on the outcome and (ii) genes belonging to no cluster - omitting a large number of irrelevant contributions. Microarrays measure expression for the entire genome in one experiment, but genes may change expression, independent of the experimental condition (e.g. due to stage in cell cycle). Forced inclusion (as demanded by complete clusterings) in well-defined but inappropriate groups may impact the final structures found for the data. Partial clustering avoids the situation where an interesting sub-group in a cluster is obscured through forcing membership of unrelated genes.

Finally, clusterings can be categorized as *exclusive (hard)*, or *overlapping*. Exclusive clustering requires each gene to belong to a single cluster, whereas overlapping clusterings permit genes to simultaneously be members of numerous clusters. An additional sub-categorization of overlapping can be made between *crisp* and *fuzzy* membership. Crisp membership is boolean - either the gene belongs to a group or not. With fuzzy membership values, each gene belongs to each cluster with a membership weight between 0 (absolutely doesn't belong) and 1 (absolutely belongs). Clustering algorithms which permit genes to belong to more than one cluster are typically *more applicable* to GE since: (i) the impact of noisy data on clusters obtained is reduced - the assumption is that "noisy" genes are unlikely to belong to any one cluster but are equally likely to be members of several clusters, (ii) the underlying principal of clustering GE, is that genes with similar change in expression for a set of samples are involved in a similar biological function. Typically, gene products are involved in several such biological functions and groups need not be co-active under all conditions. This gives rise to high variability in the gene groups and/or some overlap between them. For these reasons, constraining a gene to a single cluster is counter-intuitive with respect to natural behavior.

2 Clustering Prerequisites

Lack of *a priori* knowledge means that *unsupervised* clustering techniques, where data are unlabeled or not annotated, are common in GE work. Clustering GE usually involves the following basic steps [3]:

(1) *Pattern representation:* Involves the representation of the data matrix for clustering, can refer to the number, type, dimension and scale of GE profiles available. Some of these were set during experiment execution, however, certain features are controllable, such as scaling of measurements, imputation, normalisation techniques, representations of up/down-regulation etc. An optional step of *feature selection* or *feature extraction* can be carried out. These are two distinct procedures, the former refers to selecting a subset of the original features which would be most effective to use in the clustering procedure, the latter to the use of transformations of the input features to produce new salient features that may be more discriminatory in the clustering procedure, e.g. Principal Component Analysis.

(2) *Definition of pattern proximity measure:* Usually measured as a distance between pairs of genes. Alternatively, conceptual measures can be used to characterize the similarity among a group of gene profiles e.g. Mean Residue Score of Cheng and Church (see Section 3).

(3) *Clustering or grouping the data:* The application of the clustering algorithm to find structures (clusterings) in the dataset. Clustering methods can be broadly categorised according to the taxonomy due to [3].

(4) *Data abstraction:* Representation of structures found in the dataset. In GE data, this is usually human orientated, so data abstraction must be easy to interpret (vital for follow up analysis and experimentation). It is usually a compact description of each cluster, through a cluster prototype or representative selection of patterns within the cluster, such as cluster centroid.

(5) *Assessment of output:* Validity of clustering results is essential to cluster analysis of GE data. A cluster output is valid if it cannot reasonably be achieved by chance or as an artifact of the clustering algorithm. Validation is achieved by careful application of statistical methods and testing

hypotheses. These measures can be categorised as: (i) Internal validation, (ii) External validation and (iii) Relative validation.

This survey concentrates its review on points 2 and 3, but it should be noted that these are part of a larger analysis cycle.

3 Clustering Methods

Requirements for analysis of large GE data-sets are relatively new, although pattern recognition of complex data is well-established in a number of fields, e.g. database management [12], computer vision [13], analysis of financial markets [14] among others. Many common generic algorithms have, in consequence, been drafted into use for microarray data, (e.g. Hierarchical [15], SOM's [16]), but not all perform well. A good method should deal with noisy high dimensional data, be insensitive to the order of input, have moderate time and space complexity (allow increased data load without breakdown or requirement of major changes), require few input parameters, incorporate meta-data knowledge (i.e. deal with range of attributes) and produce results, interpretable in the biological context.

3.1 Pattern Proximity Measures

All clustering algorithms use some measure of proximity to evaluate how much a group of gene vectors show coherent expression. The choice of which is as important as the choice of clustering algorithm, and is somewhat based on the type of data and context of the clustering. Many clustering algorithms work directly on a proximity matrix (e.g. hierarchical clustering) or a measure is used to evaluate a clustering during execution of the algorithm (e.g. K-Means). A proximity measure can be calculated between a pair of genes (e.g. Euclidean distance) or between a group of genes (e.g. Mean Residue Error).

Commonly used is the notion of distance, which typically measures the distance between a pair of genes. Distance functions between two vectors include the Minkowski measures (Euclidean, Manhattan, Chebyshev). These distances are useful when searching for *exact* matches between two expression profiles in the dataset, tending to find globular structures, and works well when the dataset contains compact and isolated structures. A drawback of these is the largest feature dominates the others, thus sensitive to outliers or spurious measurements [3]. The *Mahalanobis distance* differs from the Euclidean distance in that it takes into account the correlations of the dataset and is scale-invariant.

Use of one distance measure becomes problematic when clusters within the data differ in shape, e.g Euclidean distance only finds spherical clusters, while Mahalanobis finds ellipsoidal clusters. Specifically, different distance measures should be used to find clusters of different geometric shape. This idea was developed by [17], which describes an *adaptive distance norm* method for clustering data (Gaustafson-Kessel method). Here different distances are used by estimating the data co-variance of each cluster (via eigenvalue calculations) to guess the structure of the data in each individual cluster. Each cluster is then created using a unique distance measure.

Correlation distances, on the other hand, measure how much two GE profiles show similar changes in expression across samples, without regard to the scale of the expression. For example, if a gene X is up-regulated in a certain number of samples, and gene Y is correspondingly down-regulated, the correlation distances would evaluate these to be very close, and cluster them together, unlike the Minkowski distances. Correlation measures include standard *Pearson correlation coefficient*, and *cosine correlation coefficient*, and two non-parametric measures *Spearman's rank correlation* and *Kendall's* τ, used when outliers and noise is known to be in the data. Correlation measures are transformed to distance measures by subtracting from 1, and are squared when the sign of the correlation is not of interest.

Alternatively, conceptual measures of similarity can be used to evaluate a cluster. Such measures include models based on *constant rows, constant columns* and *coherent values (additive or multiplicative model)* [11] (Fig. 1), where a sub-matrix is evaluated to ascertain how well it fits a model. A good fit indicates high correlation within the sub-matrix, thus a possible cluster.

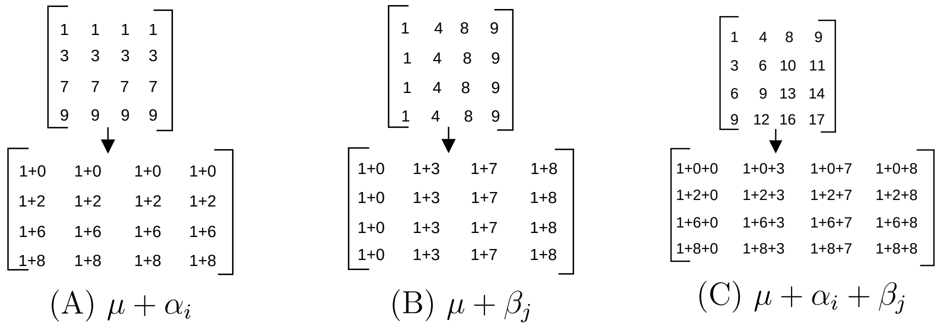

Fig. 1. Models for bi-clusters: (A) Bi-cluster with constant rows, where each row in the bi-cluster can be obtained from a typical value μ and a row offset α_i, (B) Constant columns, each value can be obtained from a typical value μ and a column offset β_j, (C) Additive model, each value can be predicted from a typical value μ, and a row and column offset, $\alpha_i + \beta_j$. These models can be adapted for the multiplicative case for (A(i)) $\mu \times \alpha_i$, (B(i)) $\mu \times \beta_j$ and (C(i)) $\mu \times \alpha_i \times \beta_j$

These models are used by a number of the clustering algorithms. For example, Cheng and Church [18] and FLOC [19], use the additive model (Fig. 1(C)), as a basis for the Mean Residue Score to evaluate bi-clusters. Given a GE matrix A, the residue of an element a_{ij} in a sub-matrix (I, J) is given by the difference $r_{ij} = (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})$ where a_{ij}, a_{iJ}, a_{Ij} and a_{IJ} are the sub-matrix value, the row, column and group mean respectively and the *"H-score"* of the sub-matrix is the sum of the residues, given by:

$$H(I, J) = \frac{1}{\mid I \mid\mid J \mid} \Sigma_{i \in Ij \in J} (r_{ij})^2 \tag{1}$$

The *Plaid Model* [20] variant builds the GE matrix as a sum of layers, where each layer corresponds to a bi-cluster. Each value a_{ij} is modelled by $a_{ij} = \sum_{k=0}^{K} \theta_{ijk} \rho_{ik} \kappa_{jk}$ where K is the layer (bi-cluster) number, ρ_{ik} and κ_{jk} are binary variables that represent the membership of row i and column j in layer k. For layer k, expression level θ_{ijk} can be estimated using the general additive model, $\theta_{ijk} = \mu_k + \alpha_{ik} + \beta_{jk}$, where μ_k is the typical value of a gene *in layer* k, α_{ik} is the row adjustment for row i and β_{jk} is the column adjustment for column j, *in layer k.* (Fig. 1 (C))

Coherent Evolutions model can be used to evaluate a cluster. In this manner, the exact values of x_{ij} are not directly taken into account, but a cluster is evaluated to see if it shows coherent patterns of expression. In its simplest form, each value can have three states: up-regulation, down-regulation and not change. Care must be taken to define thresholds between the states. Complexity can be added to the model to define more states such as "slightly" up-regulated, "strongly" up-regulated etc. This is the method adopted by SAMBA [21].

Other measures which evaluates the coherency of a group of genes includes *conditional entropy*: $H(C|X) = -\int \sum_{j=1}^{m} p(c_j|x) logp(c_j|x)p(x)dx$ measures uncertainty of random variable C (cluster category) on average when random variable X is known. The optimal partition of the GE dataset is obtained when this criterion is minimised i.e. each gene is assigned to only one cluster with a high probability, [22]. Of course, this requires the estimation of the *a posteriori* probabilities $p(c_j|x)$, which can be achieved via non-parametric methods (avoiding assuming the distribution of the underlying GE data), [22].

The similarity measures described above do not make any distinction between time-series data and expression measures obtained from two or more phenotypes. Applying similarity measures to time series data should be approached with care. Gene expression time series are usually very short (4-20 samples: classical analysis of time series data considers under 50 observations too short for statistical significance) and unevenly sampled. Similarity in time series should be only be viewed as similar patterns in the direction of change across time points and the measure should be able to handle: (1) scaling and shifting problems, (2) unevenly distributed sampling points and (3) shape (internal structure - times series cannot be regarded as independently, identically distributed data). For an excellent review on similarity in time series data see [23].

Algorithms shall be examined in the next section, however it should be borne in mind that each technique *shall* use some proximity measure, which will enhance/detract from the method according to that measure's properties.

3.2 Agglomerative clustering

All agglomerative techniques naturally form a hierarchical cluster structure in which genes have a crisp membership. Eisen et al. [15] implemented hierarchical clustering to study the GE in the budding yeast *Saccharmyces Cerevisiae.* Subsequently, the method has become popular in GE analysis, mainly due to its ease of implementation, visualisation capability and availability in most analysis packages. These methods vary with respect to distance metric used and the decision on cluster merging (linkage). Choice of parameters affects the structure of and relationship between the clusters. Linking methods include: *single linkage* (cluster separation as distance between two nearest objects), *complete linkage* (as previously, but two furthest objects), *average linkage* (average distance between all pairs), *centroid* (distance between centroid's of each cluster) and *Ward's* method (minimizes ANOVA Sum of Squared Errors between two clusters)[24].

The distance metric and linkage method determine level of sensitivity to noise; linkage methods, such as Wards and Complete are particularly affected, (due to ANOVA basis and weight given to outliers respectively, since the clustering decision depends on maximum distance between two genes). Single linkage, however, forces cluster merge, based on minimum distance between two genes, and regardless of the position of other genes in the clusters. Thus noisy or outlying values are among the last to be clustered. Nevertheless, one result is the "chaining phenomenon", Fig. 2, illustrated by the rectangle within cluster A. If cluster B has not been processed as a cluster in its own right, the next data point may be the closest dot, (see arrow). Average and Centroid linkage, the most popular choices, essentially resolve the problem; no special consideration is given to outliers and the cluster is based on highest density.

While the dendrogram tree is an intuitive interface to present results, there are 2^{n-1} different linear orderings consistent with the structure of the tree. This should be carefully considered when cutting the branches of the tree to determine cluster membership. Cutting the tree at one level may miss out on important information at other levels. [25] presents a method dendrogram

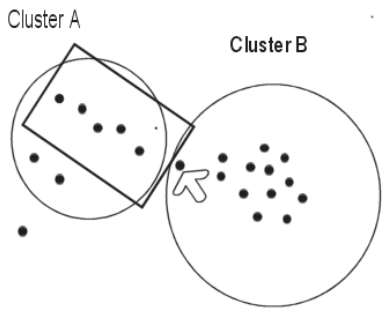

Fig. 2. Chaining phenomenon when using single linkage hierarchical clustering

analysis, based on gene class information from specialised databases. This method looks for optimal correlations between gene classes and clusters from different branch lengths. Groups with the highest statistical significance from varying branch lengths are then presented as clusters. [26] presents an agglomerative technique, which produces a k-ary tree (were each internal node has at most k children), helping to reduce the effects of noise. This allows up to k genes (or clusters) to be directly related, differing from traditional hierarchical clustering in that groups of up to k are joined together. A permutation test is used to decide on the number of nodes (max k) to merge. This involves a user specified parameter, α which will dictate the level of similarity required for a merge. Using heuristics, [26], the complexity of the algorithm is comparable to traditional hierarchical clustering. [26] also present a divide and conquer approach to create an optimal leaf ordering of the tree (order the tree leaves that maximize the sum of similarities of adjacent leaves). However, this is at increased time and space complexity, and is only feasible for small values of k.

These methods cannot correct for the greedy nature of the traditional algorithm, where mis-clusterings made at the beginning cannot be corrected and are greatly magnified as the process continues. [27] and [28] note that the performance of hierarchical clustering method is close to random, despite its wide usage and is worse than other common techniques such as K-means and Self Organising maps (SOM).

3.3 Partitive Techniques

Partitive clustering techniques divide data into clusters depending on similarity measures. Widely used methods measure distance from a gene vector to a prototype vector representing the cluster, maximising intra-cluster distance whilst minimising inter-cluster distance (e.g. *K-means*, *Fuzzy C-Means*, *SOM*). A major drawback of techniques in this category is that the number of clusters in the data must be specified beforehand, although methods have been developed to try to overcome this. Table 3 summarises algorithms presented here.

K-means produces crisp clusters with no structural relationship between them, [29]. It deal poorly with noise, since genes *must* belong to a cluster and noisy values have a substantial effect on the cluster mean, thus affecting the inclusion of other genes. Equally, the average value used for cluster inclusion of a gene, depends on values already present, so *order* of data inclusion matters. It relies greatly on the choice of initial prototypes of the clusters (different runs will result in different clusterings, thus it's unstable), and often results in a local minima solution. Incremental approaches have been shown to find local minima solutions close to the global solution, such as *Modified Global K-means (MGKM)* algorithm [30], which computes k-partitions of the data using $k-1$ clusters from the previous iterations. The starting point for the k^{th} cluster is computed using an auxiliary function. The number of clusters to search for does not need to be specified directly, but a tolerance threshold must be set which determines the number of clusters indirectly. As with regular K-means, this algorithm will search for spherical clusters. On the six datasets tested in the study, the MGKM algorithm showed only slight improvement over K-means, however at a higher computational time cost.

Algorithms which modified K-means to produce fuzzy clusters include *Fuzzy C-Means (FCM)*[31] and *Fuzzy clustering by Local Approximations of MEmberships (FLAME)* [32]. Genes are assigned a membership degree to a clustering indicating its percentage association with that cluster. The two algorithms differ in the weighting scheme used for the contribution of a gene to the mean of the cluster. FCM membership values for a gene is divided among clusters in proportion to similarity with that clusters mean. The contribution of each gene to the mean of a cluster is weighted, based on its membership grade. Membership values are adjusted iteratively until the variance of the system falls below a threshold. These calculations require the specification of a *degree of fuzziness* parameter which is problem specific [31]. In contrast FLAME (available with the GEDAS software suite) membership of a gene to a cluster, i, is determined by a weighted similarity to its K-nearest neighbours, and the memberships of these neighbours to cluster i. These values are updated in an iterative process. In both techniques, genes may belong to several clusters, with varied degree of membership. However, membership grades from the two techniques have different interpretations. FCM and FLAME uses an averaging techniques, where each gene contributes to the calculation of a cluster centroid, typically memberhship values must sum to 1, i.e. gene-cluster probability, so a strong membership does not indicate that a given gene has a more typical value, and thus need careful interpretation [33].

Table 2 illustrates this idea for three clusters. FCM was carried out on published yeast genomic expression data [34], results available at `http://rana.lbl.gov/FuzzyK/data.html`. The membership values for gene B and gene D

Table 1
Difficulties interpreting membership values of FCM

GID	**Cluster 4**		**Cluster 21**		**Cluster 46**	
	Centroid Dist.	Mem.	Centroid Dist.	Mem.	Centroid Dist.	Mem.
GENE1649X	10.691	0.002575	8.476	0.002002	3.864	0.482479
GENE6076X	6.723	0.009766	3.855	0.009341	6.33	0.007381
GENE5290X	6.719	0.007653	5.29	0.00515	8.024	0.005724
GENE2382X	7.725	0.007609	3.869	0.01782	6.279	0.010249

Table 2
GENE1649X (Gene A), GENE6076X (Gene B), GENE5290X (Gene C) and GENE2382X (Gene D). The table highlights distance to cluster centroid, as calculated by Euclidean distance, and the associated membership values of the gene.

are very different for cluster 21, although they are approximately equidistant from the centroid of the cluster. Similarly gene C and gene D have comparable membership values for cluster 4, however gene C is more typical than gene D. With similar centroid distance measures, membership value for gene B in cluster 21 is smaller than membership value of gene A in cluster 46. These arise from the constraint on membership values, which must sum to one across all clusters, forcing a gene to give up some of its membership in one cluster to increase it in another. Listing the genes of a cluster based on membership values is non-intuitive as it is not a measure of their compatibility with the cluster. However, if analysing the list as the degree of sharing between clusters then it is correct. This effect is similar for FLAME, as the memberships are weighted only by the K-nearest neighbours, a low membership value indicates a high degree of cluster sharing among its K-nearest neighbours and not a more typical value. [35] recognised this interpretative flaw, and developed the *possibilistic biclustering* algorithm, which removes the restriction that membership values must sum to unity. These authors used spectral clustering principals [36] to create a partition matrix, Z, of the original GE matrix. Z is then clustered using a possibilistic approach. The resulting clusters are evaluated using the H-Score (Eq.1), and showed improvements over more traditional techniques. This algorithm requires the specification of two parameters amongst others-cutoff membership for a gene to be included in a cluster and a cutoff membership for a sample to be in included in a cluster. However, in this case these parameters make sense as they indicate how typical a gene/sample is to a cluster, and *not* its degree of sharing.

The membership weighting system reduces noise effects, as a low membership grade is less important in centroid calculation. By assuming that noise and outlier genes are dissipated across all clusters by the weighting, relative relationships are not distorted by a few extreme values. The density based approach of FLAME (density estimated from similarity to K-nearest neighbours) reduces this further, where genes with a density lower than a pre-defined threshold are categorised as outliers, and grouped with a dedicated 'outlier' cluster. As with K-Means, clusters produced by FCM are unstable, largely

	Cluster Mem.	**Input**	**Proximity**	**Other**
K-Means	Binary	Starting Prototypes, Stopping Threshold, K	Pairwise Distance	Very Sensitive to input parameters and order of input.
MGKM	Binary	Tolerance Threshold	Pairwise Distance	Not as sensitive to starting prototypes. K specified through tolerance threshold.
FCM	Fuzzy	Degree of fuzziness, Starting prototypes, Stop threshold, K	Pairwise Distance	Careful Interpretation of membership values. Sensitive to input parametres and order of input.
FLAME	Fuzzy	K_{nn} - number of neighbours	Pairwise Distance to K_{nn} neighbours	careful interpretation of membership values. Output determined by K_{nn}.
Possibilistic biclustering	Fuzzy	Cut-off memberships Max. residue, number of rows and number of columns	H-Score	Number of biclusers determined when quality function peaks by re-running for different numbers of eignevalues.

Table 3
Summary of Partitive techniques. With the exception of FLAME and Possibilistic biclustering, all find complete global clusters. Note: Sections 3.4 - 3.7 highlight subcategorisations of partitive techniques.

affected initialisation parameters and the number of clusters must be decided in advance. FLAME produces stable clusters, however the size of the neighbourhood and the weighting scheme used will affect results and the number of clusters.

3.4 Evolutionary approaches

Evolutionary approaches (EA) are motivated by natural evolution and make use of evolution operators (selection, recombination, mutation) and a population of possible solutions (chromosomes) to obtain a global optimal partition of the data. A fitness function gives the chromosomes likelihood of surviving until the next generation. Although evolutionary approaches attain the global optimum, as ever, these approaches suffer from sensitivity to user defined input parameters and these must be tuned for each specific problem. Genetic algorithms (GAs) are an EA approach which work well when the dataset is considered small (less than 1000 gene vectors and of low dimension), above which the time constraints are prohibitory, thus an unlikely choice for GE analysis. *GeneClust* [37] used the GA approach, where the fitness function corresponds to minimising the *Total Within Cluster Variance (TWCV)* of the clusters produced. The datasets chosen for testing were of small size (all under 900 gene expression profiles), and only showed comparative results with K-means (known to find local minima) and Click (see below) and Cast (see below).

Local minimum solutions found by K-means is linked with the initial seed selection. If a good initial partition can be obtained quickly using other techniques, then it may work well. There has been studies which combine the powers of K-means and Genetic Algorithms. [38] developed *Incremental Genetic K-Means Algorithm (IGKA)*, a hybrid approach which converges to a global optimum faster than stand alone GA, without the sensitivity to initialization prototypes. The fitness function for the GA is based on the *TWCV*. The basic idea behind the algorithm is to cluster centroids incrementally, using a standard similarity measure. Of course, the GA method requires the number of output clusters needs to be specified, in addition to the difficult to interpret mutation probability rates, the number of generations and the size of the chromosome populations. These last three parameters will influence how long the algorithm will take to converge to a global optimum. GA are not limited to TWCV, but could be used to minimise a range of fitness functions.

3.5 Neural Networks

Neural Networks (NN) are based on biological nervous system, i.e. nerves and synaptic connections. Modelled as a collection of nodes with weighted interconnections, NNs only process numerical vectors, so meta-information cannot be included in the clustering procedure. They learn their interconnection weights adaptively i.e. acting as feature selectors by appropriate selection of weights.

Self Organising Maps (SOMs) are a popular type of NN enabling visualisation and interpretation oh high dimensional data [16], and have been widely used in analysis of GE data [39, 40, 41]. A SOM comprises of an input layer of N nodes (each node represents a dimension of a gene) and an output layer of neurons arranged in a regular 1d or 2d grid. Each neuron is represented by a reference (or weight) vector of the same dimension as the input gene. A kernel function defines the region of influence (neighbourhood) that the input gene has on the SOM. This kernel function update causes the reference vector of the output neuron and its neighbours to track to-wards the gene vector. The network is trained (to adjust strengths of interconnections) from a random sample of the dataset. Once training is complete, all genes in the dataset are then applied to the SOM, where only one output neuron 'fires' upon receiving an input signal from a gene, (hard clustering). Cluster members, represented by output neuron i are the set of genes, applied to the input neurons, causing output neuron i to 'fire'.

SOM's are robust to noise and outliers, with results still somewhat effected, dependent on distance metric and the neighbourhood function used. If the training data contains noise its effect is exacerbated by over-fitting. Like K-means, SOM produces a sub-optimal solution if the initial weights for the interconnections are not chosen properly. Its convergence to a solution is controlled by problem specific parameters such as learning rate and neighbourhood function; a particular input pattern can fire different output neurons at different iterations. This can be overcome by gradually reducing the learning rate to zero during training. However, this can cause the network to overfit to the training data, and not preform well when presented with new data. Again, the number of clusters must be specified beforehand, via specifying the number of output neurons; too few output neurons in the SOM gives large within-cluster distance, too many results in meaningless diffusion.

The *Self Organizing Tree Algorithm (SOTA)* [42], *Dynamically Growing Self Organizing Tree (DGSOT)* algorithm [43] and, more recently, *Growing Hierarchical Tree SOM (GHTSOM)* [**?**] were developed to incorporate NN strengths (i.e. speed, robustness to noise) with those of hierarchical clustering (i.e. tree structure output, minimum requirement to specify number of clusters a priori, no training required). Here the SOM network is a tree structure, trained by comparing only the leaf nodes to the input GE profiles (each node in the graph represents a cluster). At each iteration the leaf node with the highest degree of heterogeneity is split, creating two daughter cells, in the case of SOTA, and dynamically finding the proper number of daughters, in the case of DGSOT (determines the number of daughters by starting off with two and continually adding one until certain cluster validation criteria are satisfied). To determine the correct number of daughter cells [43] propose a method based on the geometric characteristics of the data i.e. based on cluster separation in the minimum spanning tree of the cluster centroids. A threshold must be specified by the user, (a practical consideration, arrived at through trial and error (the authors propose 0.8)). Growth of the tree continues until the heterogeneity of the system falls below a threshold, or until all genes map onto a unique leaf node. This has the advantage over most partitive techniques in that the number of clusters is not specified before hand, indirectly by the threshold as to when to stop growing. In [42]the threshold is determined via a re-sampling technique, where system variability is defined as the maximum distance among genes mapping to the same leaf node. By comparing all possible distances between randomized data and those between the real dataset, a confidence interval and a distance cut-off are obtained. DGSOT method uses average leaf distortion to determine when to stop growing, so the user indirectly specifies the number of clusters via this threshold.

In the case of GHTSOM, each node is a triangular SOM (SOM with 3 neurons, fully connected), each having 3 daughter nodes (also triangular SOMs). Growth in this system occurs if a neuron is activated by a sufficient number of inputs, i.e. at least 3 or a user defined variable which will determine the resolution of the system. Growth continues as long as there is one neuron in the system which can grow.

After each addition of a new daughter a training process is performed. SOTA, DGSOT and GHTSOM differ from the typical hierarchical clustering algorithm, in that once a gene is mapped to a leaf node, it and its neighbours are adapted, as in the SOM, however the neighbourhood of the adaptation is a lot more restrictive. In SOTA if it maps to a leaf node with a sister node, the latter, the ancestor node and the winning node are all adapted. If there is no sister node, then only the winning node is adapted. Similarly for DGSOT, however this incorporates a method to correct for misclusterings at the beginning of the process. This involves a K-Level up distribution (KLD), where data incorrectly clustered when clustering starts can be reclustered during later hierarchical growing stages. This involves specification of K - the ancestor level in the tree of the node which is to be created into two daughter cells. DGSOT then distributes all values among the leaves of the subtree rooted at the K^{th} ancestor, overcoming the misclusterings problem of the traditional hierarchical algorithm, SOTA and GHTSOM, although it does present yet another input parameter for the user to specify. These method incorporates a pairwise distance mechanism to calculate distances between profiles, so properties of this measure will affect the results. In GHTSOM, new nodes (after growth) are trained using only the inputs which caused the parent node to fire. The standard SOM training algorithm is used to train the new nodes, followed by a fine tuning training phase. This involves specification of the learning rate parameters by the user, as with traditional SOMs. Finally any neuron which shows low activity is deleted, and its parent is blocked from further growth. This has the advantage that inputs mapping to leaf neurons at the top levels of the hierarchy are usually noise, distinct from relevant biological patterns. At each hierarchical level, clusters of neurons are created by presenting the layer with the entire training dataset and finding the winning neuron (smallest Euclidean distance between input and neuron). Links are then created between the winning neuron and a set of neighbors, i.e. neurons which fall within a range determined by the output (Euclidean distance) of the winning and losing nodes in the sub-network. Leaf nodes are considered separately and linked to the neuron with the smaller response for each input that activated that leaf.

SOTA and DGSOT successful combines two different algorithms to deal with the properties of gene expression data, in that the output shows a hierarchical structure, with specified resolution, ameliorating interpretation compared to traditional partitive techniques. Additionally, the cluster number depends on data variability, and not number of data points.

	Structure	**Proximity**	**Input**	**Other**
SOM	None	Distance	number of output neurons, Learning rate	Careful consideration of initalisation weights
SOTA	Binary Tree	Distance	Threshold t	
DGSOT	Tree	Distance	Thresholds t, α and k.	Corrects for misclusterings
GHTSOM	SOMs in Tree struct.	Distance	Minimal requirement- learning rate	

Table 4
Summary of Neural Network techniques presented.

3.6 Search Based

Search techniques obtain a solution for a criterion function by searching the solution space either deterministically or stochastically [3]. As deterministic techniques exhaustively search all solutions, these would be of no use for gene expression analysis. Typically, heuristics are used to guide these searches.

A well known stochastic search method is *Simulated Annealing*, applied to GE analysis by [44, 45]. [44] uses the *TWCV* as the fitness function, E, to minimise, where as [45] applied it to minimise the *H-Score)* 1. At each stage it the process, gene vectors are randomly chosen and moved to a new random cluster. The fitness function is evaluated for each move and if it improves, then the new assignment is accepted, otherwise the new assignment is accepted with a probability of $e^{-\frac{E^{new}-E^{old}}{T}}$, where T is temperature, and controls how readily the system will accept the worse situation, enabling the algorithm to escape local minima. As the search continues, T, is gradually reduced according to an *annealing schedule*, in the end the search only accepts better situations, and will reach a global minimum. The parameters annealing schedule will dictate the performance and speed of the search. This involves setting an initial temperature - if set to high the system will take to long to converge, if set to low the potential search space will be too much reduced. Similarly for the final effective temperature (when to stop searching for a solution). Additionally, the user must specify the rate at which T approaches the final effective temperature - it must be reduced slowly enough to guarantee a global minimum. The user must also specify how many swaps of gene vectors between clusters is allowed during each iteration.

To determine the optimal number, [44] uses a randomisation procedure to determine cut-off threshold for the distance, D, between two gene vectors in a single cluster. The problem of identifying the optimal number of clusters, becomes a problem of determining parameters D and P, the probability of accepting false positives, which can be set to a reasonable value (e.g. $P =$

0.05). Once P and D have been determined, the process involves re-running the simulated annealing procedure for different numbers of clusters, until the weighted average fraction of incorrect gene vector pairs reaches the P-value.

Cheng and Church [18] adapted the method of Hartigan [46] to GE data. Their algorithm proceeds by scoring sub-matrices of the GE matrix using the H-score (Eq. 1, Fig. 1) [11]. A *perfect bi-cluster* would have a H-score equal to *zero* (corresponding to ideal GE data, with matrix rows and columns constant or additive). This method considers sub-matrix a bi-cluster if $H(I, J) < \delta$ for some $\delta \geq 0$ (user defined).

Once a sub-matrix is determined to be a bi-cluster, the values of this sub-matrix are masked with random numbers in the initial GE matrix. Masking bi-clusters prevents the algorithm from repeatedly finding the same sub-matrices on successive iterations. However, there is a substantial risk that this replacement will interfere with the discovery of future bi-clusters, [19]. To overcome this problem of random interference, *Flexible Overlapped biClustering (FLOC)* was developed - a generalised model of Cheng and Church which incorporates null values, [19]. FLOC constrains the clusters to a low mean residue score *and* a minimum occupancy threshold of α, which is a positive number ≤ 1 (user defined). Note: this method does not require pre-processing for imputation of missing values.

These bi-clustering algorithms find coherent groups in the data and permit overlapping. [20] indicated that the value of an element in the GE matrix can be modelled as a linear function of the contributions of the different bi-clusters to which the row i and the column j belong, (Fig. 3),[20].

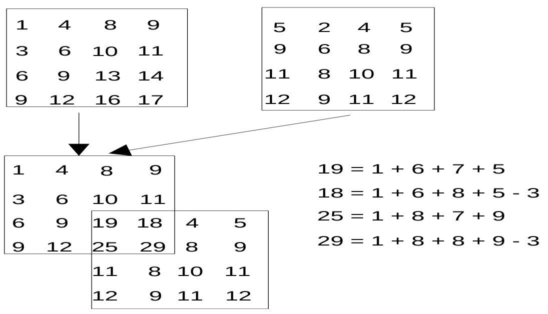

Fig. 3. With Plaid Model values at overlaps are seen as a linear function of different bi-clusters.

The Plaid Model [20] assumes that bi-clusters can be generated using a statistical model and tries to identify the distribution of the parameters that best fit the available data by minimising the sum of squared error for the K^{th} bi-cluster assuming that $K-1$ bi-clusters have already been identified. Explicitly, to minimize: $Q = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m}(Z_{ij} - \theta_{ijk}\rho_{ik}\kappa_{jk})^2$ for the entire matrix, where Z_{ij} is

	Proximity	**Deterministic/ Stochastic**	**Clusters**	**Other**
SA	Depends on application	Stochastic	Depends on application	Specification of Annealing Schedule
CC	Additive Model	Deterministic	Overlapping, partial bi-clusters	δ, random interference
FLOC	Additive Model	Deterministic	Overlapping, partial bi-clusters	α and δ to specify. Overcomes random interference, allows missing values.
Plaid	Additive Model	Deterministic	Overlapping, partial bi-clusters	Values seen as sum of contributions to bi-clusters

Table 5
Summary of search based techniques presented.

the residual after deducting the previous layers found: $Z_{ij} = a_{ij} - \sum_{k=0}^{K-1} \theta_{ijk}\rho_{ik}\kappa_{jk}$

The parameters θ_{ijk}, ρ_{ik} and κ_{jk} are estimated for each layer and for each value in the matrix, and are themselves updated in an iterative approach, alternately estimating parameters μ_k (Eq. **??**), α_{ik} and β_{jk} and parameters ρ_{ik} and κ_{jk} in an effort to minimise Q, [20].

The importance of a layer is defined by $\delta_k^2 = \sum_{i=1}^{n} \sum_{j=1}^{p} \rho_{ik}\kappa_{jk}\theta_{ijk}^2$. To evaluate the significance of the residual matrix, Z is randomly permuted and tested for importance. If δ_k^2 is significantly better than δ_{random}^2, k is reported as a bi-cluster. The algorithm stops when the residual matrix Z retains only noise, with the advantage that the user does not need to specify the number of clusters beforehand.

3.7 Graph Theoretic Methods

Graph theoretic approaches have become popular in analysing large complex datasets. A GE dataset can be modelled as a graph, $G = (V, E)$, where V is the set of vertices, representing genes or samples and E is the set of edges connecting vertices. Alternatively, a bi-partite graph structure can be used, $G = (U, V, E)$, where U is the set of sample vertices and $U \cap V = \varnothing$ and an edge (u, v) only exists between $v \in V$ and $u \in U$. The definition of an edge is dependant on the algorithm used. In the methods outlined below, a complete GE graph is partitioned, where a cluster corresponds to a subgraph, and the cluster number and the cluster structure is determined from the data. In the output clustering there is no structural relationship between the clusters. A major strength of the techniques outlined below, is that probabilistic models are incorporated into the assigning of edge weights. Thus there is a statistical significance associated with the output clusters.

Cluster Affinity Search Technique (CAST) [47] models the data as an undirected graph, where the vertices represent genes, and an edge indicates similar expression pattern. The model assumes that there is an ideal *clique* graph (a disjoint union of complete sub-graphs), $H = (U, E)$, which represents the ideal input gene expression dataset and data to be clustered is obtained from ideal graph H by "contamination" with random errors. A clique graph represents a possible clustering of the gene expression dataset, where each clique represents a cluster. The model assumes that a pair of genes where incorrectly assigned similarity/dissimilarity with a probability of α and that the input graph was generated from H by flipping each edge, non-edge with a probability of α. The true clustering of the input graph is assumed to be those that imply fewer flips. CAST uses an affinity (similarity) measure to assign a vertex to a cluster, this can be binary (1 = affinity, 0 otherwise), or real valued, measured by some distance function. The affinity to a cluster must be above a threshold, t, which is defined by the user. This, of course, will influence the size and the number of clusters. The affinity of a vertex v with a cluster, is the sum of affinities with all objects currently in the cluster. v has high affinity with i if $affinity_(x) > t|i|$, and low affinity otherwise. The CAST algorithm alternates between adding high affinity elements and removing low affinity elements, finding clusters one at a time. The result is dependant on the order of input as once a clustering structure is obtained, for every vertex v, if there is another cluster for which v has a higher affinity value, it is moved to this cluster.

CLICK [48], builds on the work of [49], which uses a probabilistic model for the edge weights. Initalized with a similarity measures between all pairs of gene vectors, this model assumes pairwise similarity measures between genes are normally distributed: between 'mates' with mean μ_T and variance σ_T^2, and between 'non-mates' with μ_F and σ_F^2, where $\mu_T > \mu_F$. These parameters can be estimated via Expectation Maximisation methods. The weight of an edge is derived from the similarity measure between the two gene vectors, and reflects the probability that $i \in V$ and $j \in V$ are mates, specifically: $w_{ij} = log\frac{p_{mates}\sigma_F}{(1-p_{mates})\sigma_T} + \frac{(S_{ij}-\mu_F)^2}{2\sigma_F^2} - \frac{(S_{ij}-\mu_T)^2}{2\sigma_T^2}$

Edge weights below a user defined non-negative threshold t are omitted from the graph. In each step the algorithm handles some connected component of un-clustered elements, partitioning the graph according to a minimum weight cut.

The *Statistical Algorithmic Method for Bi-cluster Analysis (SAMBA)* method finds bi-clusters based on the coherent evolution model (Section 3.1) [21]. Firstly, the gene expression matrix is modelled as a bipartite graph, $(u, v) \in E$ iff there is a significant change in expression level of gene v, w.r.t. to the gene's normal level, under condition u. A key idea for SAMBA is the scoring scheme for a bi-cluster, corresponding to its statistical significance, where a weight is assigned to a given edge, (u, v), based on the log-liklihood of getting that weight by chance [21], $log\frac{P_c}{P_{(u,v)}} > 0$ for edges and, $log\frac{(1-P_c)}{(1-P_{(u,v)}} < 0$ for non-edges, where the probability $P_{(u,v)}$ is the fraction of random bipartite graphs with degree sequence identical to G that contain edge (u, v) (and can be estimated using Monte-Carlo methods), and P_c is a constant probability $> max_{(u,v)\in UxV} P_{(u,v)}$. Assigning these weights to the edges and non-edges in the graph, the statistical significance of the subgraph H can be calculated simply by its weight. After assigning weights to the edges and non-edges in the graph, the K heaviest (largest weight) sub-graphs for each vertex in G are found.

Two models are presented for calculating the weight of the resulting sub-graph. In the first and simpler model, bi-clusters are sought which manifest changes relative to their normal level without considering if this corresponds to an increase or decrease in expression,[21]. The weight of the sub-graph $H(U', V', E')$ is taken as a sum of all the edge and non-edge weights. In the second model, the search focus is *consistent bi-cliques*, targeting conditions, which affect genes in a consistent way, (e.g. changes in which two conditions must have the same or opposite effect on each of the genes).

The *heaviest* sub-graphs are found by scanning all subsets of gene v's neighbours, returning the sub-graph for gene v with the largest weight. SAMBA identifies bi-clusters through an iterative enumeration of all possible bi-clusters, guiding the search by heuristics, such as restricting the degree of the gene vertices, i.e. considering only genes with less than d incident edges, and considering only limited subsets of neighbouring vertices for a particular gene vertex.

Expression Data Clustering Analysis and Visualisation Resource (EXCAVATOR) is a suite of tools developed by [50] for the identification of subtrees of the Minimum Spanning Tree of a gene expression graph. Each subtree represents a cluster. These subtrees do no overlap, thus producing a complete, crisp partition of the dataset. This tool provides three clustering algorithms and all use some distance measure to weight the edges of the tree, thus depending on the distance function used different clusters will be produced. The first one is very simple, which finds a user specified number of clusters by cutting the branches of the longest length (heaviest weight = furthest distance). This works well as long as the inter cluster distance is greater than the intra cluster distance. The second one minimises the variance within a cluster. As it compares to a mean value, it tends to find globular clusters, and local minimum. The third method uses representatives of the cluster and minimises on this, to find globally optimum clusters. This allows for the identification of non-globular clusters, as a representative may closer reflect a clusters data points rather than a centre. When the clusters are globular, the representative

	Mode	**Proximity**	**Search**	**Other**
CAST	One Mode	Similarity	Clique Graph	Parameters α and t. Finds global, complete, crisp clusters.
CLICK	One Mode	Distribution based on distance	Minimum weight cut	Stat. Sig. of clusters. EM to estimate parameters. Finds global, partial, crisp clusters.
SAMBA	Bi-Partite	Probability	Heuristic search of neighbours	Stat. sig. of clusters. Input P_c difficult to define. Finds partial overlapping bi-clusters.
EXCAVATOR	Tree	Similarity	Minimum Spanning Tree	Complete, global, crisp partition. Must specify the number of clusters.

Table 6
Summary of Graph theoretic methods presented.

is generally close to the centre.

4 Discussion

Despite the shortcomings, the application of clustering methods to GE data has proven to be of immense value, providing insight on cell regulation, as well as on disease characterisation. Nevertheless, not all clustering methods are equally valuable in the context of high dimensional gene expression. Recognition that well-known, simple clustering techniques, such as K-Means and Hierarchical clustering, do not capture more complex local structures in the data, has led to bi-clustering methods, in particular, gaining considerable recent popularity. Indications to date are that these methods provide increased sensitivity at local structure level for discovery of meaningful biological patterns.

An inherent problem with exploratory clustering is the unknown number of groups in advance of clustering. This has been identified by most creators of algorithms and have included methods in their techniques which try to determine this from the data at run time. However, care must be taken not to replace this input parameter with complicated thresholds which may not make sense in the context of gene expression data. Principally, selection of the correct *number* of clusters (K) typically requires assessment of several different potential values and validation of clusters produced. The correct K will minimise heterogeneity between, while maximising homogeneity within groups. Determining the number of *significant* clusters includes, not only direct extraction (or assessment) of clusterings, but appropriate hypothesis testing. Direct methods use criteria such as cluster sum of squares[8], likelihood ratios[51], average silhouette[52] or Mean Split Silhouette[53]. Tests include Gap Statistic[54], Weighted Discrepant Pairs (WADP)[55] and a variety of permutation methods[55], [9]. Nevertheless, improvement identification of non-global clusters is low and since most involve bootstrapping, these methods can be very computationally expensive. Those methods which do not need the number of clusters specified before execution of the algorithm have an advantage.

A survey to give justice to the range of robust methods of validation and evaluation of clusterings is not possible due to space limitations. See [56] for a comprehensive review of these methods. Incorporation of these techniques must be realised to achieve full potential of clustering methods. A number of clustering techniques have been developed which integrate clustering information with meta information, [57, 58, 59, 60, 61, 62], such as information from the Gene Ontology database [63]. These methods integrate information from clustering with gene ontology information for external validation and simplifies the interpretation of clustering results.

5 Conclusion

Cluster analysis applied to microarray measurements aims to highlight meaningful patterns for gene co-regulation. The evidence suggests that, while commonly used, agglomerative and partitive techniques are insufficiently powerful given the high dimensionality and nature of the data. Bi-clustering methods, although computationally expensive, are readily interpretable in terms of data features and structure. While further testing on non-standard and diverse data sets is required, the comparative assessment and numerical evidence to date support the view that bi-clustering methods have considerable potential for analysis of gene expression data. The limitations of commonly used algorithms have been well documented in microarray analysis literature, and new methods have been created and adapted from existing algorithms to overcome these. The creators of these hybrid methods must take care not to replace the limitations of existing techniques with complex parameters which are difficult to determine in the gene expression context. However, the uptake of new methods by the user community has been slow, mainly due to their algorithmic complexity. This would be hastened by increased transparency of the algorithm and availability in analysis packages.

References

[1] M. H. Asyali, D. Colak, O. Demirkaya, and M. S. Inan. Gene expression profile classification: A review. *Curr. Bioinformatics*, 1(1):55–73, 2006.

[2] T. D. Pham, C. Wells, and D. I. Crane. Analysis of microarray gene expression data. *Curr. Bioinformatics*, 1(1):37–53, 2006.

[3] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM CSUR*, 31(3):264–323, 1999.

[4] S. O. Zakharkin, K. Kim, T. Mehta, L. Chen, S. Barnes, K. E. Scheirer, R. S. Parrish, D. B. Allison, and G. P. Page. Sources of variation in affymetrix microarray experiments. *BMC bioinformatics*, 6:214, Aug 29 2005.

[5] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman. Missing value estimation methods for dna microarrays. *Bioinformatics (Oxford, England)*, 17(6):520–525, Jun 2001.

[6] A. G. de Brevern, S. Hazout, and A. Malpertuy. Influence of microarray experiments missing values on the stability of gene groups by hierarchical clustering. *BMC bioinformatics*, 5:114, Aug 23 2004.

[7] X. Liu, G. Cheng, and J. X. Wu. Analyzing outliers cautiously. *IEEE Trans. Knowl. Data Eng.*, 14(2):432–437, 2002.

[8] G. W. Milligan and M. C. Cooper. An examination of procedures for determining the number of clusters in a dataset. *Psychometrika*, 50:159–179, 1985.

[9] J. Fridlyand and S. Dudoit. Applications of resampling methods to estimate the number of clusters and to improve the accuracy of a clustering method. Technical Report 600, Department of Statistics, University of California, Berkeley, 2001.

[10] A. Ben-Dor, B. Chor, R. Karp, and Z. Yakhini. Discovering local structure in gene expression data: the order-preserving submatrix problem. *J. Comput. Biol.*, 10(3-4):373–384, 2003.

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

[12] S. Guha, R. Rastogi, and K. Shim. Cure: an efficient clustering algorithm for large databases. In *SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 73–84, New York, NY, USA, 1998. ACM Press.

[13] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: theory and its application to image segmentation. *IEEE TPAMI*, 15(11):1101–1113, 1993.

[14] M. Marsili. Dissecting financial markets: sectors and states. *Quant. Finance*, (4):297–302, 2002.

[15] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci USA*, 95(25):14863–14868, Dec 8 1998.

[16] T. Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[17] D. W. Kim, K. H. Lee, and D. Lee. Detecting clusters of different

geometrical shapes in microarray gene expression data. *Bioinformatics*, 21(9):1927–1934, May 1 2005.

[18] Y. Cheng and G. M. Church. Biclustering of expression data. *ISMB '00*, 8:93–103, 2000.

[19] J. Yang, H. Wang, W. Wang, and P. Yu. Enhanced biclustering on expression data. *BIBE '03*, page 321, 2003.

[20] L. Lazzeroni and A. B. Owen. Plaid models for gene expression data. *Statistica Sinica*, 12(1):61–86, 2000.

[21] A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, 18 Suppl 1:S136–44, 2002.

[22] H. Li, K. Zhang, and T. Jiang. Minimum entropy clustering and applications to gene expression analysis. *Proc. IEEE CSB.*, pages 142–151, 2004.

[23] C. Moller-Levet, K. H. Cho, H. Yin, and O. Wolkenhauer. Clustering of gene expression time-series data. Technical report, 2003.

[24] A. Sturn. Cluster analysis for large scale gene expression studies, 2001.

[25] P. Toronen. Selection of informative clusters from hierarchical cluster tree with gene classes. *BMC bioinformatics*, 5:32, Mar 25 2004.

[26] Z. Bar-Joseph, E. D. Demaine, D. K. Gifford, N. Srebro, A. M. Hamel, and T. S. Jaakkola. K-ary clustering with optimal leaf ordering for gene expression data. *Bioinformatics (Oxford, England)*, 19(9):1070–1078, Jun 12 2003. LR: 20061115; PUBM: Print; JID: 9808944; ppublish.

[27] K. Y. Yeung, D. R. Haynor, and W. L. Ruzzo. Validating clustering for gene expression data. *Bioinformatics*, 17(4):309–318, Apr 2001.

[28] F. D. Gibbons and F. P. Roth. Judging the quality of gene expression-based clustering methods using gene annotation. *Genome Res.*, 12(10):1574–1581, Oct 2002.

[29] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In L. M. Le Cam and J. Neyman, editors, *Proc. Fifth Berkeley Symp. Math. Stat. Prob.*, pages 281–297. University of California Press, 1967.

[30] Adil M. Bagirov and Karim Mardaneh. Modified global k-means algorithm for clustering in gene expression data sets. In *WISB '06*, pages 23–28, Darlinghurst, Australia, 2006. Australian Computer Society, Inc.

[31] D. Dembele and P. Kastner. Fuzzy c-means method for clustering microarray data. *Bioinformatics (Oxford, England)*, 19(8):973–980, May 22 2003.

[32] L. Fu and E. Medico. Flame, a novel fuzzy clustering method for the analysis of dna microarray data. *BMC bioinformatics*, 8:3, Jan 4 2007.

[33] R. Krishnapuram and J. M. Keller. A possibilistic approach to clustering. *IEEE TFS*, 1(2):98–110, 1993.

[34] A. P. Gasch and M. B. Eisen. Exploring the conditional coregulation of yeast gene expression through fuzzy k-means clustering. *Genome biology*, 3(11):RESEARCH0059, Oct 10 2002.

[35] C. Cano, L. Adarve, J. Lopez, and A. Blanco. Possibilistic approach for biclustering microarray data. *Comp. Biol. Med.*, 37(10):1426–1436, Oct 2007.

[36] Y. Kluger, R. Basri, J. T. Chang, and M. Gerstein. Spectral biclustering of microarray data: coclustering genes and conditions. *Genome research*, 13(4):703–716, Apr 2003.

[37] V. Di Gesu, R. Giancarlo, G. Lo Bosco, A. Raimondi, and D. Scaturro. Genclust: a genetic algorithm for clustering gene expression data. *BMC bioinformatics*, 6:289, Dec 7 2005.

[38] Y. Lu, S. Lu, F. Fotouhi, Y. Deng, and S. J. Brown. Incremental genetic k-means algorithm and its application in gene expression data analysis. *BMC bioinformatics*, 5:172, Oct 28 2004.

[39] P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E. S. Lander, and T. R. Golub. Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation. *Proc. Natl. Acad. Sci. USA*, 96(6):2907–2912, Mar 16 1999.

[40] P. Toronen, M. Kolehmainen, G. Wong, and E. Castren. Analysis of gene expression data using self-organizing maps. *FEBS letters*, 451(2):142–146, May 21 1999.

[41] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, Oct 15 1999.

[42] J. Herrero, A. Valencia, and J. Dopazo. A hierarchical unsupervised growing neural network for clustering gene expression patterns. *Bioinformatics (Oxford, England)*, 17(2):126–136, Feb 2001.

[43] F. Luo, L. Khan, F. Bastani, I. L. Yen, and J. Zhou. A dynamically growing self-organizing tree (dgsot) for hierarchical clustering gene expression profiles. *Bioinformatics*, 20(16):2605–2617, Nov 1 2004.

[44] A. V. Lukashin and R. Fuchs. Analysis of temporal gene expression profiles: clustering by simulated annealing and determining the optimal number of clusters. *Bioinformatics*, 17(5):405–414, May 2001.

[45] K. Bryan, P. Cunningham, and N. Bolshakova. Application of simulated annealing to the biclustering of gene expression data. *IEEE T-ITB*, 10(3):519–525, Jul 2006.

[46] J. A. Hartigan. Direct clustering of a data matrix. *JASA*, 67(337):123–129, 1972.

[47] A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. *J. Comput. Biol.*, 6(3-4):281–297, Fall-Winter 1999.

[48] R. Sharan and R. Shamir. Click: a clustering algorithm with applications to gene expression analysis. *ISMB '00*, 8:307–316, 2000.

[49] E. Hartuv, A. O. Schmitt, J. Lange, S. Meier-Ewert, H. Lehrach, and R. Shamir. An algorithm for clustering cdna fingerprints. *Genomics*, 66(3):249–256, Jun 15 2000.

[50] Y. Xu, V. Olman, and D. Xu. Clustering gene expression data using a graph-theoretic approach: an application of minimum spanning trees. *Bioinformatics*, 18(4):536–545, Apr 2002.

[51] A. J. Scott and M. J. Symons. Clustering methods based on likelihood ratio criteria. *Biometrics*, 27(2):387–397, Jun. 1971.

[52] L. Kaufmann and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis.* John Wiley and Sons Inc, Chinchester New York Weinheim, 1990.

[53] M. J. van der Laan and K. S. Pollard. Hybrid clustering of gene expression data with visualization and the bootstrap. Technical Report U.C. Berkeley Division of Biostatistics Working Paper 93, 2001.

[54] R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a data set via the gap statistic. *J Royal Statistical Soc B (Statistical Methodology)*, 63(2):411–423, 2001.

[55] M. Bittner, P. Meltzer, Y. Chen, Y. Jiang, E. Seftor, M. Hendrix, M. Radmacher, R. Simon, Z. Yakhini, A. Ben-Dor, N. Sampas, E. Dougherty, E. Wang, F. Marincola, C. Gooden, J. Lueders, A. Glatfelter, P. Pollock, J. Carpten, E. Gillanders, D. Leja, K. Dietrich, C. Beaudry, M. Berens, D. Alberts, and V. Sondak. Molecular classification of cutaneous malignant melanoma by gene expression profiling. *Nature*, 406(6795):536–540, Aug 3 2000.

[56] J. Handl, J. Knowles, and D. B. Kell. Computational cluster validation in post-genomic data analysis. *Bioinformatics (Oxford, England)*, 21(15):3201–3212, Aug 1 2005. LR: 20061115; PUBM: Print-Electronic; DEP: 20050524; JID: 9808944; RF: 71; 2005/05/24 [aheadofprint]; ppublish.

[57] G. Gamberoni, S. Storari, and S. Volinia. Finding biological process modifications in cancer tissues by mining gene expression correlations. *BMC Bioinformatics*, 7(6):8, Jan 9 2006.

[58] R. Kustra and A. Zagdanski. Incorporating gene ontology in clustering gene expression data. *cbms*, 0:555–563, 2006.

[59] J. Kasturi and R. Acharya. Clustering of diverse genomic data using information fusion. *Bioinformatics*, 21(4):423–429, 2005.

[60] C. Yang, E. Zeng, T. Li, and G. Narasimhan. Clustering genes using gene expression and text literature data. In *CSB '05*, pages 329–340, Washington, DC, USA, 2005. IEEE Computer Society.

[61] J.K. Choi, J.Y. Choi, D.G. Kim, D.W. Choi, B.Y. Kim, K.H. Lee, Y.I. Yeom, H.S. Yoo, O.J. Yoo, and S. Kim. Integrative analysis of multiple gene expression profiles applied to liver cancer study. *FEBS Letters*, 565(1-3):93–100, 2004.

[62] J. Liu, W. Wang, and J. Yang. Gene ontology friendly biclustering of expression profiles. In *CSB '04*, pages 436–447, Washington, DC, USA, 2004. IEEE Computer Society.

[63] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris,

D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock. Gene ontology: tool for the unification of biology. the gene ontology consortium. *Nat. Genet.*, 25(1):25–29, May 2000.