Identifying Projected Clusters from Gene Expression Profiles

Kevin Y. Yip[1*], David W. Cheung[1], Michael K. Ng[2] and Kei-Hoi Cheung[3]

[1] Department of Computer Science and Information Systems, University of Hong Kong, Hong Kong

[2] Department of Mathematics, University of Hong Kong, Hong Kong

[3] Center for Medical Informatics and Department of Genetics, Yale University School of Medicine, New Haven, Connecticut, USA

[*] Corresponding author
Address: Department of Computer Science and Information Systems, University of Hong Kong, Pokfulam Road, Hong Kong.
Fax: (852)2559-8447
Email: dcheung@csis.hku.hk

ABSTRACT

In microarray gene expression data, clusters may hide in certain subspaces. For example, a set of co-regulated genes may have similar expression patterns in only a subset of the samples in which certain regulating factors are present. Their expression patterns could be dissimilar when measuring in the full input space. Traditional clustering algorithms that make use of such similarity measurements may fail to identify the clusters. In recent years a number of algorithms have been proposed to identify this kind of projected clusters, but many of them rely on some critical parameters whose proper values are hard for users to determine. In this paper a new algorithm that dynamically adjusts its internal thresholds is proposed. It has a low dependency on user parameters while allowing users to input some domain knowledge should they be available. Experimental results show that the algorithm is capable of identifying some interesting projected clusters.

Keywords: gene expression, projected clustering, data mining.

1. INTRODUCTION

Clustering is a popular data mining technique for extracting information from gene expression profiles. A large variety of clustering methods have been used to generate many kinds of interesting clusters. Some recent studies include [9, 10, 15, 20]. The goal of these clustering methods is to partition similar objects (samples or genes) into clusters such that intra-cluster similarity is maximized while inter-cluster similarity is minimized. Sample clustering is common in tumor studies for identifying tumor subtypes [4, 14, 21]. Gene clustering has been used to predict groups of genes that have similar functions or are co-regulated [8, 13, 17]. It has also become very popular to cluster both samples and genes individually and visualize the results in a single figure [4]. In this paper, we will use the terms *object* and *dimension* to mean a row and a column of a dataset respectively. An object refers to a gene when performing gene clustering, and refers to a sample when performing sample clustering. The opposite holds for a dimension.

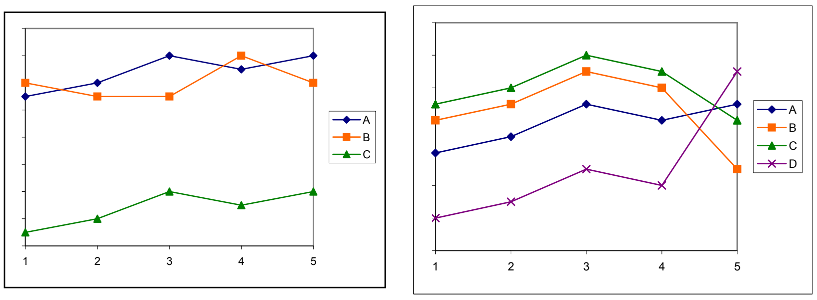
All the mentioned methods assume object similarity is measured in the input space formed by all the dimensions of a dataset. It has been pointed out that gene expression data may exhibit some checkerboard structures [18, 23], in which each block is defined by a subset of objects and a subset of dimensions where the objects are similar when considering only the dimensions in the subset. When all dimensions are considered, the objects may appear to be dissimilar. This may occur when, for example, two genes have similar expression patterns only in a subset of samples where certain regulating factors are present. In the other samples, the two genes may express differently. Each block can be viewed as a cluster of objects “projecting” onto a subspace defined by the corresponding dimensions. This kind of clusters is thus referred to as *projected clusters* [1].

A similar concept has long been studied in supervised learning. For example, in decision tree classifiers [25], each generated rule can be regarded as a region in a subspace that contains mostly the members of a single class. We consider in this paper the unsupervised version of the problem, which concerns the finding of both object partitions and their subspaces from unlabeled data. Also, we assume that all members of a cluster can be found in a single region in a subspace (cf. multiple decision rules may be needed to cover all objects of a class).

We now give a more formal definition of projected clusters. Given a dataset D with N objects and a set V of d input dimensions, a *projected cluster* C_I contains N_I objects and is defined in a d_I-dimensional subspace formed by the set V_I of dimensions, where $V_I \subseteq V$. In the subspace, the members of C_I are similar to each other according to a similarity function, but dissimilar to other objects not in C_I. d_I is called the *dimensionality* of cluster C_I, which is the size of the set of *relevant dimensions* V_I of the cluster. The complementary set $V - V_I$ is called the *irrelevant dimensions* of the cluster. The members of a cluster are dissimilar in the subspace formed by its irrelevant dimensions. A dimension can be relevant to zero, one, or more clusters. To distinguish the clusters defined based on some domain knowledge and the clusters identified by a clustering algorithm, we will call the former ones the *real clusters* and their relevant dimensions the *real relevant dimensions*, while the latter kind of clusters will simply be called the *clusters* and the identified relevant dimensions the *selected dimensions*. In the literature the term "class" is commonly used to represent a group of objects defined according to some domain knowledge. The dimensions or attributes that describe each class are called its "features". In the current text, sometimes we need to represent a set of objects in a dataset that belong to the same class. Since the set is a sample of the class instead of the class itself, we prefer to use the less popular term "real cluster" to describe it instead of calling it a "class" to avoid confusion.

Notice that the definition does not assume any kind of object similarity, although a cluster is most often regarded as a group of objects having a small distance from each other (based on a distance function such as Euclidean distance). This kind of clustering, what we will describe as *distance-based*, has been successful in many studies on gene expression data analysis. For instance, most of the studies cited above implicitly assume distance-based clustering. In this paper we will also introduce a new algorithm that is distance-based. On the other hand, there are situations where it is more suitable to measure the similarity between two objects by their rise and fall expression patterns [7, 19, 27]. Two objects are similar if they have the same direction of response across the relevant dimensions, regardless of their absolute expression values. We will discuss later how this kind of *pattern-based clustering* can be handled by a modified distance-based clustering algorithm.

The above concepts are illustrated in Figure 1. In part a, distance-based clustering assumes A is more similar to B than to C, while pattern-based clustering assumes the reverse. Part b shows a projected cluster based on pattern-based clustering, to which dimension 5 is irrelevant.



(a) Distance-based clustering vs. pattern-based clustering.

(b) Relevant dimension vs. irrelevant dimension.

Figure 1: An example illustrating some discussed concepts. In (a), A is more similar to B than to C in distance-based clustering, but it is more similar to C than to B in pattern-based clustering. In (b), the objects have similar value trends except along dimension 5, so dimension 5 is an irrelevant dimension of the cluster.

The goal of a projected clustering algorithm is to form a number of high-quality projected clusters. Basically, a cluster is of high quality if its member objects are unexpectedly similar. The actual quality measure used in this paper will be described in Section 3. We will first assume clusters are disjoint, i.e., each object belongs to only one cluster, and later extend our study to consider also non-disjoint clusters since they are common in gene clustering, where each gene may belong to multiple groups according to different categorizations.

Before moving on, we need to emphasize the difference between projected clustering and feature selection. Although both concern the selection (and possibly construction) of important features, feature selection defines a feature space for the whole dataset, while projected clustering identifies a possibly different subspace for each cluster. Due to the difference, feature selection is performed prior to clustering, while a projected clustering algorithm determines the subspace of each cluster during the clustering process. Feature selection can be used as a preprocessing step of projected clustering, but it alone cannot solve the projected clustering problem.

The remaining of this paper is organized as follows: in the next section, we will review some projected clustering approaches proposed in recent years, and discuss some of their potential problems. A new algorithm will be proposed in Section 3, which is designed to avoid the problems. Experimental results on real datasets will be presented in Section 4, and some discussions and the conclusion of the study will be given in Section 5 and Section 6 respectively.

2. RELATED WORK

There have been a lot of studies on projected clustering and its related problems *subspace clustering* [3] and *biclustering* [7] in recent years. A thorough survey of the three problems can be found

	Cluster definition	Disjoint clusters	Clusters reported
Projected clustering	Distance-based	True	Highest quality
Subspace clustering	Distance-based	False	All passing quality thresholds
Biclustering	Mainly pattern-based	False	Highest quality

Table 1: Comparing the three related problems.

in [29]. Table 1 summarizes their main differences. In this section we focus on the related work on the projected clustering problem, which assumes a distance-based similarity definition and produces disjoint clusters. We are especially interested in this problem because of the large number of fruitful studies on clustering gene expression profiles that also make the two assumptions, and the few reported studies on applying projected clustering on gene expression profiles.

There are two major challenges in projected clustering that make it distinctive from traditional clustering. The first challenge is the simultaneous determination of both cluster members and relevant dimensions. Cluster members are determined by calculating object distances in the subspace formed by the relevant dimensions, while the relevant dimensions are determined by measuring the distances between the projections of the cluster members along different dimensions. One common approach to tackling this chicken-and-egg problem is to form some tentative clusters according to some heuristics, determine their relevant dimensions, and then refine the cluster members based on the selected dimensions. The heuristics being used are critical to the effectiveness of the algorithm. If inappropriate heuristics are used, the tentative clusters formed will not help the discovery of real clusters.

The second challenge is determining the dimensionality of each cluster, which is usually unknown to users when working on gene expression profiles due to the lack of domain knowledge and the large number of possible values given the high dimensionality of data.

We now review some proposed projected clustering approaches and discuss how they are affected by the challenges. The partitional approach PROCLUS [1] is based on the k-medoids method [22]. As in traditional k-medoids methods, some objects are initially chosen as the medoids. But before assigning every object in the dataset to the nearest medoid, each medoid is first temporarily assigned a set of neighboring objects that are close to it in the input space to form a tentative cluster. For each tentative cluster, all dimensions are sorted according to the average distance between the projected values of the medoid and the neighboring objects. On average l dimensions with the smallest average distances are selected as the relevant dimensions for each cluster, where l is a parameter value supplied by user. Normal object assignment then resumes, but the distance between an object and a medoid is computed using only the selected dimensions. Medoids with too few assigned objects are regarded as outliers, which are replaced by some other objects to start a new iteration.

The user parameter l may introduce a usability problem since the correct value to use is hard to determine. Another potential problem arises when the real clusters have few relevant dimensions, in which case the cluster members may not be close to each other in the full input space. Since the tentative clusters are formed based on distance calculations in the input space, when a member of a real cluster is chosen as a medoid, the neighboring objects assigned to it may not come from the same real cluster. Subsequently, the dimensions selected would not be the real relevant dimensions and the resulting cluster would be mixed of objects from different real clusters.

Another partitional algorithm ORCLUS [2] was proposed to improve PROCLUS. According to the experimental results reported in [2], it is more accurate and stable than PROCLUS. Nevertheless, it still relies on user-supplied values in deciding the number of dimensions to select for each cluster.

In the hypercube approach DOC and its variant FastDOC [24], each cluster is defined as a hypercube with width 2ω, where ω is a user supplied value. The clusters are formed one after another. To find a cluster, a pivot point is randomly chosen as the cluster center and a small set of objects is randomly sampled to form a tentative cluster around the pivot point. A dimension is selected if and only if the distance between the projected values of every sample and the pivot point on the dimension is no more than ω. The tentative cluster is thus bounded by a hypercube with width 2ω. All objects in the dataset falling into the hypercube are grouped to form a candidate cluster. More random samples and pivot points are then tried to form more candidate clusters, and a specially designed function is used to evaluate the quality of them. The candidate cluster with the best evaluation score is accepted, and the whole process repeats to find other clusters.

As with PROCLUS and ORCLUS, the selected dimensions of DOC and FastDOC are determined by a user parameter. In addition, they also restrict each cluster to be a hypercube with equal width along all relevant dimensions, which is unlikely to be true in real data. Tentative clusters are formed by random sampling, which avoids direct distance calculations in the input space. However, the number of tentative clusters required to try can become so large that seriously affects the speed performance.

Summarizing the above observations, in order to apply projected clustering on gene expression data, it would be preferable to develop an algorithm that can identify the dimensionalities of the clusters directly from data and avoid the formation of problematic tentative clusters. In the next section we will describe a new projected clustering algorithm HARP (a Hierarchical approach with Automatic Relevant dimension selection for Projected clustering) [29] that satisfies these requirements. It is an agglomerative hierarchical clustering algorithm with each object treated as a singleton cluster at the beginning, and the most similar clusters are merged iteratively according to a merge score. Three building components of the algorithm will be introduced first, followed by a description of the complete algorithm, its computational complexity, and some possible extensions.

3. THE HARP ALGORITHM

3.1 Relevance Index, Cluster Quality and Merge Score

In distance-based projected clustering, a cluster can be viewed as a group of objects being unexpectedly close to each other in a certain subspace. In other words, for a dimension to be relevant to a cluster, the projections of the cluster members on the dimension should be unexpectedly close to each other. This closeness can be measured by the ratio of the variance within the cluster to the variance in the whole dataset. Denote σ^2_{Ij} as the variance of projected values of all objects in C_I along dimension v_j (the *local variance*) and $\sigma^2_{\cdot j}$ as the variance of projected values along v_j in the whole dataset (the *global variance*), the *relevance index* of v_j in cluster C_I is defined as follows:

$$R_{Ij} = 1 - \frac{\sigma^2_{Ij}}{\sigma^2_{\cdot j}}. \tag{1}$$

The index gives a high value when the local variance is small compared to the global variance, which refers to the situation where the projections of the cluster members on the dimension are close, and the closeness is not due to a small average distance between the projected values in the whole dataset. A dimension receives an index value close to the maximum of one if the local variance is extremely small, which means the projections form an excellent *signature* for identifying the cluster members. Alternatively, if the local variance is only as small as the global variance, the dimension will receive an index value of zero. This suggests a baseline for dimension selection: a negative R value indicates a dimension is not more relevant to a cluster than to a random sample of objects. The dimension should therefore not be selected. We will discuss later how this baseline is used to define the stopping criteria of HARP.

To prevent the index from being undefined in some degenerate situations, we assume there does not exist any dimensions with zero global variances (on which all objects have the same projected value). If such a dimension does exist, it would not be useful at all and could be safely removed before the clustering process. Also, if a cluster contains only one object, the index values of all dimensions are set to one.

Conceptually, incorporating the global variance in the relevance index is similar to performing normalization to the dataset. The use of the index thus implicitly performs normalization without the need of an explicit preprocessing step. An advantage of the index is the strong intuitive meaning of the sign of its values, which helps interpret the clustering results.

Based on the relevance index, the quality of a cluster C_I can be measured by the sum of the index values of all the selected dimensions:

$$Q_I = \sum_{v_j \in V_I} R_{Ij}. \tag{2}$$

In general, the more selected dimensions a cluster has, and the larger are their respective R values, the larger will be the value of Q. We define the quality measure in this way since an identified cluster is more likely to consist of objects from the same real cluster (the cluster is more likely to be "correct") if the identified cluster has more selected dimensions and the dimensions have higher relevance index values [29]. We will discuss how HARP determines the relevant dimensions of each cluster later. At this point it can be assumed that each cluster has a reasonable set of selected dimensions.

Similarly, a score can be defined to evaluate the merge between two clusters. Basically, if two clusters can be merged to form a cluster with a high quality, the merge is a potentially good one, i.e., the two clusters probably contain objects from the same real cluster. However, in case the two merging clusters have a large size difference, an unfavorable situation called *mutual disagreement* can occur. Consider a large cluster with a thousand objects and a small one with only five objects. If they are merged to form a new cluster, its mean and variance of projected values will highly resemble the original values of the large cluster, which will dominate the choice of the dimensions to be selected. If a dimension is originally selected by the large cluster, it will probably be selected by the new cluster also no matter the projected values of the small cluster are close to those of the large cluster or not. The resulting cluster can have a high Q score even the two clusters have a strong mutual disagreement on the signatures of the resulting cluster.

To cope with this problem, we modify the relevance index to take into account the mutual disagreement phenomenon. Suppose C_{I_3} is the resulting cluster formed by merging C_{I_1} and C_{I_2}, the mutual-disagreement-sensitive relevance index of dimension v_j in C_{I_3} is defined as follows:

$$\begin{aligned}
R^*_{I_3j} &= \frac{R_{I_1j|I_2} + R_{I_2j|I_1}}{2}, && (3) \\
R_{I_1j|I_2} &= 1 - \frac{\sigma^2_{I_1j} + (x_{I_1j} - x_{I_2j})^2}{\sigma^2_{\cdot j}} \\
&= 1 - \frac{\sum_{x_i \in C_{I_1}} (x_{ij} - x_{I_2j})^2 / N_i}{\sigma^2_{\cdot j}}, && (4)
\end{aligned}$$

where x_{ij} is the projection of object x_i on dimension v_j, and x_{Ij} is the mean projected value of all members of cluster C_I on v_j. $R_{I_1j|I_2}$ is the adjusted relevance index of v_j in C_{I_1} given that C_{I_1} is merging with C_{I_2}. The numerator of its second term is the average squared distance between the projected values of C_{I_1} on v_j from the mean projected value of C_{I_2}. $R_{I_2j|I_1}$ is defined similarly. If the two clusters do not agree on the values along v_j, $(x_{I_1j} - x_{I_2j})^2$ will effectively diminish the R^* score of the dimension. The original R index is used to determine the quality of a cluster, while the modified index R^* is used to determine the merge score between two clusters. When C_{I_1} and C_{I_2} are merged to form C_{I_3}, the merge score is as follows:

$$\begin{aligned} MS(C_{I_1}, C_{I_2}) &= \sum_{v_j \in V_{I_3}} R^*_{I_3 j} \\ &= \sum_{v_j \in V_{I_3}} \frac{R_{I_1 j|I_2} + R_{I_2 j|I_1}}{2} \\ &= \sum_{v_j \in V_{I_3}} [1 - \frac{\sigma^2_{I_1 j} + \sigma^2_{I_2 j} + 2(x_{I_1 j} - x_{I_2 j})^2}{\sigma^2_{\cdot j}}]. \end{aligned} \tag{5}$$

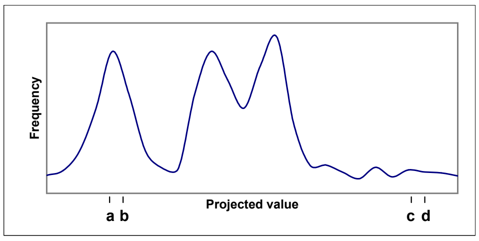
The MS score will be used to determine the merge order. Merges with higher MS scores will be allowed to perform earlier.

3.2 Validation of Similarity Scores

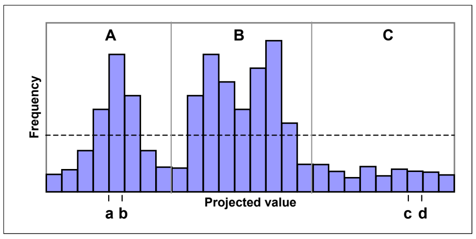
The MS function concerns both the quality and number of selected dimensions, but does not take into account the size of a cluster. Suppose there is a set C of objects all belonging to real clusters to which dimension v_j is irrelevant. If the size of C is small, it is not uncommon to find the projections of the objects in C on v_j being close to each other due to random chance. If C is large, the probability for the same phenomenon to occur is relatively small. Looking in another way, if a dimension has a high relevance index value in a cluster, the more objects the cluster contains, the less likely the high index value is merely by chance.

Since HARP is a hierarchical algorithm with each initial cluster containing a single object, it is not meaningful to incorporate cluster size directly in the calculation of merge scores. However, it is possible to utilize the *potential* cluster size in estimating the significance of a cluster, which can be obtained from the frequency distribution of projected values. Figure 2a shows the distribution of a typical dimension that is relevant to some real clusters. The distribution contains a number of peaks, which correspond to the signatures of the real clusters. The base level at the troughs is likely due to random values. Suppose a cluster contains members with projected values within the interval $[a, b]$, it has a high potential to merge with other clusters to form a cluster with a significant size and a high concentration of projected values around the $[a, b]$ region. On the other hand, if a cluster contains members with projected values within the interval $[c, d]$, although the cluster may receive a high R score at the dimension, the cluster is unable to keep the high R value if it is to grow to a significant size. In other words, the high concentration of projected values is probably due to random chance. The R value of the cluster should therefore be invalidated in order to prevent more objects to be merged into the cluster due to the fake signature.

Based on the observation, we develop a histogram-based validation mechanism for preventing the formation of incorrect clusters due to the fake signatures. The idea is that if a dimension is relevant to a cluster, the corresponding histogram should contain a peak around the signature values



(a) The frequency distribution.



(b) A histogram built from the distribution.

Figure 2: The frequency distribution of a typical dimension. If a dimension is relevant to some clusters, its frequency distribution should contain a number of peaks much higher than the random noise level, which is the average bin frequency (dotted line in (b)) in case of a uniform background distribution.

(see regions A and B in Figure 2b). The width and height of the peak depend on the properties of the cluster, but provided the cluster has a significant size, the peak should exceed the random noise level, which corresponds to the mean frequency in case of a uniform distribution (shown by the dotted line). Clusters covered by bins that stay below the noise level are statistically insignificant (region C), and the relevance index value of the dimension in the cluster will be set to zero.

The histogram-based validation is usually applied on gene clustering only, but not on sample clustering. This is because in the latter case the number of objects (samples) is usually too small to build a histogram that could simulate the real distribution of expression values.

3.3 Dynamic Threshold Loosening

When we introduced the MS function in Section 3.1 we assumed that there is a way to determine the relevant dimensions of each cluster. In this section we discuss how this is made possible by the dynamic threshold loosening mechanism.

As discussed in Section 3.1, a cluster is more likely to be correct if it contains a larger number of selected dimensions, and the selected dimensions have higher relevance index values. This means merges that form resulting clusters with both properties should be allowed to perform earlier. Practically, this is achieved by two internal thresholds R_{min} and d_{min}. Two clusters are allowed to merge if and only if the resulting cluster has d_{min} or more selected dimensions, and a dimension v_j is selected by C_I if and only if $R^*_{Ij} \geq R_{min}$. At any time, the two thresholds define a set of allowed merges where the actual merging order within the set is determined by the MS scores.

At the beginning, R_{min} and d_{min} are initialized to their tightest (most restrictive, i.e., highest) values 1 and d respectively. All allowed merges produce clusters that contain identical objects, so the clusters must be correct. At some point, there will be no more qualified merges. The thresholds will be slightly loosened to qualify some new merges. Whenever all qualified merges have been performed, the thresholds will be further loosened. As clustering proceeds, the clusters grow bigger in size. The projections of the cluster members on the real relevant dimensions remain close to each other, but the chance of having similar closeness of projections on other dimensions drops, so as their relevance index values. This allows the real relevant dimensions to be clearly differentiated from the irrelevant dimensions, which in turn ensures the formation of correct clusters.

In order to guarantee the quality of the final clusters, the two thresholds are associated with baseline values such that when the baselines are reached, no further loosening is allowed. As mentioned in Section 3.1, a negative R value means that a dimension is very unlikely to be relevant to a cluster. The baseline of R_{min} is thus set to zero. For d_{min}, the baseline is set to one, which is the minimum value for a cluster to be defined as a projected cluster. We will see later that the HARP algorithm allows users to specify an optional target number of clusters. According to our experience, if such a value is specified, the algorithm usually finishes the clustering process well before the thresholds reach their baselines. The clusters produced thus contain selected dimensions with R scores much better than that of a random set of projected values.

There are many possible ways to loosen the threshold values. From our empirical study, a simple linear loosening scheme is found to be very adaptive and performed well. In this scheme, there is a fixed number of threshold levels such that whenever no more qualified merges remain, the values of the two thresholds are updated using a linear interpolation towards the baseline values (see Section 3.4 for details). By default, we set the number of threshold loosening steps to the dataset dimensionality d such that after each threshold loosening, d_{min} is reduced by 1.

Obviously, while the simple loosening mechanism and the default number of loosening steps work well in our experiments, they are not always the best choice. To this end, we allow users to input some domain knowledge should they be available. Users are allowed to input the initial and baseline values for the two thresholds and the number of loosening steps. They may also select an alternative loosening scheme (e.g. aggressive loosening that always loosens the threshold that leads to more qualified merges, or conservative loosening that does the reverse), or specify their preferred scheme as a plugin procedure.

3.4 The Complete Algorithm

The skeleton of the whole algorithm using the simple loosening scheme and the default parameter values is shown in Algorithm 1, and Procedures 2 to 6 list the pseudo codes of its main procedures.

At the beginning of the clustering process, each object forms a singleton cluster. The dimensionality and relevance thresholds d_{min} and R_{min} are initialized to their tightest values. For each cluster, the dimensions that satisfy the threshold requirements are selected. The merge score between each pair of clusters is then calculated. Only merges that form a resulting cluster with d_{min} or more selected dimensions are qualified and the others are ignored.

Algorithm 1 The HARP algorithm.

```
Algorithm HARP (k: target no. of clusters (default: 1))
1   For $step$ := 0 to $d - 1$ do {
2       $d_{min} := d - step$
3       $R_{min} := 1 - step/(d - 1)$
4       Foreach cluster $C_I$
5           SelectDim($C_I$, $R_{min}$)
6       BuildScoreCache($d_{min}$, $R_{min}$)
7       While cache is not empty {
8           // $C_{I_1}$ and $C_{I_2}$ are the clusters involved in the
9           // best merge, which forms the new cluster $C_{I_3}$
10          $C_{I_3} := C_{I_1} \cup C_{I_2}$
11          SelectDimNew($C_{I_3}$, $R_{min}$)
12          UpdateScoreCache($C_{I_3}$, $d_{min}$, $R_{min}$)
13          If clusters remained = $k$
14              Goto 17
15      }
16  }
17  ReassignObjects()
End
```

procedure 2 The score cache building procedure.

```
Procedure BuildScoreCache($d_{min}$: dim. threshold, $R_{min}$: rel. threshold)
1   Foreach cluster pair $C_{I_1}$, $C_{I_2}$ do {
2       $C_{I_3} := C_{I_1} \cup C_{I_2}$
3       SelectDimNew($C_{I_3}$, $R_{min}$)
4       If $d_{I_3} \geq d_{min}$
5           Insert $MS(C_{I_1}, C_{I_2})$ into score cache
6   }
End
```

The algorithm repeatedly performs the best merge according to the MS scores of the qualified merges. In order to efficiently determine the next best merge, merge scores are stored in a cache (e.g. a quad tree or a Conga line [12]). After each merge, the scores related to the merged clusters are removed from the cache, and the best scores of the qualified merges that involve the new cluster are inserted back. The selected dimensions of the new cluster are determined by its members according to R_{min}. According to the definition of R, if a dimension is originally not selected by both merging

procedure 3 The dimension selection procedure for an existing cluster.

```
Procedure SelectDim($C_I$: target cluster, $R_{min}$: rel. threshold)
1   Foreach dimension $v_j$
2       If $R_{Ij} \geq R_{min}$ and ValidRel($C_I, v_j$)
3           Select $v_j$ for $C_I$
End
```

procedure 4 The dimension selection procedure for a new cluster.

```
Procedure SelectDimNew($C_{I_3}$: target cluster, $R_{min}$: rel. threshold)
1   Foreach dimension $v_j$ {
2       // $C_{I_3}$ is a potential cluster formed by merging $C_{I_1}$ and $C_{I_2}$
3       If $R^*_{Ij} \geq R_{min}$ and ValidRel($C_{I_1}, v_j$) and ValidRel($C_{I_2}, v_j$)
4           Select $v_j$ for $C_{I_3}$
5   }
End
```

procedure 5 The relevance index validation procedure.

```
Procedure ValidRel($C_I$: target cluster, $v_j$: target dimension)
1   lowv := max($x_{Ij} - 2\sigma_{Ij}$, $min_{Ij}$)
2   highv := min($x_{Ij} + 2\sigma_{Ij}$, $max_{Ij}$)
3   If mean frequency of the bins covering [lowv, highv] <
    mean frequency of all bins
4       return false
5   Else
6       return true
End
```

procedure 6 The score cache updating procedure.

```
Procedure UpdateScoreCache($C_{I_3}$: new cluster,
$d_{min}$: dim. threshold, $R_{min}$: rel. threshold)
1   // $C_{I_3}$ is formed by merging $C_{I_1}$ and $C_{I_2}$
2   Delete all entries involving $C_{I_1}$ and $C_{I_2}$ from cache
3   Foreach cluster $C_{I_4} \neq C_{I_3}$ do {
4       $C_{I_5} := C_{I_3} \cup C_{I_4}$
5       SelectDimNew($C_{I_5}$, $R_{min}$)
6       If $d_{I_5} \geq d_{min}$
7           Insert $MS(C_{I_3}, C_{I_4})$ into score cache
8   }
End
```

clusters, it must not be selected by the new cluster. However, if a dimension is originally selected by one or both of the merging clusters, it may or may not be selected by the new cluster.

Whenever the cache becomes empty, there are no more qualified merges at the current threshold level. The thresholds will be loosened linearly according to the formulas in lines 2 and 3 of Algorithm 1. Further rounds of merging and threshold loosening will be carried out until a target number of clusters remain, or the thresholds reach their baseline values and no more qualified merges exist.

To further improve clustering accuracy, an optional object reassignment step can be performed after the completion of the hierarchical part. The MS score between each clustered object and each cluster is computed based on the final threshold values when the hierarchical part ends. After computing all the scores, each of the objects is assigned to the cluster with the highest MS score. The process repeats until convergence or a maximum number of iterations are reached.

The parameter k that specifies the target number of clusters is optional. Like other hierarchical clustering methods, k can be set to 1 and the whole clustering process can be logged as a dendrogram[1], which allows users to determine the cluster boundaries from a graphical representation (e.g. [11]), or cut the tree according to the merge order of the clusters and a value of k determined a posteriori. Also, it can be observed that the dynamic threshold loosening mechanism relies on the hierarchical nature of HARP. These explain why we adopt the hierarchical approach in spite of its intrinsic high time complexity. HARP is especially suitable for applications where accuracy is the first priority and the datasets are of moderate sizes, such as gene expression profiles. For instance, clustering a typical gene expression dataset with 5000 genes and 50 samples takes ten to twenty minutes on a desktop PC, which is quite reasonable. In order to deal with very large datasets, we will discuss some speedup methods in the next section.

3.5 Complexity Analysis

It can be shown that the worst case time complexity of HARP is $O(N^2d^2 + Nf(N))$, where $f(N)$ is a function depending on the cache structure being used to store merge scores. For example, $f(N)$ equals N when quad tree is used and $N \log^2 N$ when Conga line is used.

It is possible to improve the speed performance of HARP in a number of ways. For two clusters to be qualified for merging, the number of common dimensions that pass the histogram-based validation must exceed the d_{min} threshold. By checking the maximum number of such common dimensions of all cluster pairs, many threshold levels could be skipped if they contain no qualified merges. This optimization is most useful when the dimensionalities of the clusters are low relative to the dataset dimensionality. Similarly, when determining the merge score between two clusters,

[1]Due to the threshold requirements, it is not always possible to merge the objects into a single cluster at the end of clustering. In general, the dendrograms of HARP are forests of trees.

the R^* value of each dimension of the resulting cluster is computed in turn. Once the number of selected dimensions is confirmed to be lower than d_{min}, the R^* values of the remaining dimensions do not need to be computed as the merges must not be qualified.

In practice, the execution time of HARP is reasonable with medium-sized datasets, but it can become unacceptable when the dataset size or dimensionality is very large. We propose two ways to speedup the clustering process. When the dataset size is large, clustering can be performed on a random sample of objects. Upon completion of the clustering process, each unsampled object is filled back to the most similar cluster subject to the restriction of the final threshold values. When the dataset dimensionality is high, a constant number of threshold levels can be used (line 2 of Algorithm 1), so that the quadratic term with respect to d in the total time complexity becomes linear.

3.6 Extensions

As discussed previously, there are situations where pattern-based clustering and non-disjoint clusters are desirable. HARP can be extended to satisfy these two requirements. To consider pattern-based similarity, the input dataset is first preprocessed by subtracting each expression value by the row average so that all resulting rows have a zero mean. Each resulting expression value measures the relative expression level of the object on the particular dimension. The distance between two preprocessed objects captures their pattern similarity in the full input space. A similar mechanism is carried out to determine the pattern similarity between two clusters in the subspace of the resulting cluster formed by merging the clusters. Suppose clusters C_{I_1} and C_{I_2} have relevant dimensions V_{I_1} and V_{I_2} respectively, and they can be merged to form C_{I_3}. The potential set of relevant dimensions of C_{I_3}, $V_{I_3}^{est}$, is estimated by the intersection of V_{I_1} and V_{I_2}. Each object in C_{I_1} and C_{I_2} subtracts their expression values by the mean expression along the dimensions in $V_{I_3}^{est}$. The distance between the two clusters in the subspace formed by $V_{I_3}^{est}$ thus captures their pattern similarity in the subspace. The set of selected dimensions can be refined by comparing the relevance index value of each dimension with the R_{min} threshold, and the process can be repeated a few times to identify a satisfactory set of selected dimensions.

When clustering completes, for each produced cluster C_I, all the objects in the dataset will be examined to see if they can be merged into C_I without lowering its quality. Each object is regarded as a singleton cluster, and its expression values are adjusted as described above according to the relevant dimensions of C_I. The MS score between it and C_I is calculated subject to the thresholds where d_{min} and R_{min} are set as the number and minimum R value of the relevant dimensions of C_I. All the objects involved in the allowed merges are assigned as members of C_I. Since each object can be assigned to multiple clusters, the final clusters are likely to be non-disjoint.

4. EXPERIMENTS

In this section we present the experimental results of HARP on three real datasets. Due to space limitation, we omit other extensive experimental results that compare HARP with seven projected and non-projected clustering algorithms on both synthetic and real datasets. The results show that HARP is able to identify some projected clusters hidden in some low-dimensional space that are missed by the other algorithms. The details can be found in [29].

4.1 Datasets

Lymphoma: It is a dataset used in studying distinct types of diffuse large B-cell lymphoma (DL-BCL)(Figure 1 of [4]). It contains 96 samples, each with 4026 expression values. The samples are categorized into 9 classes according to the category of mRNA sample studied. We used HARP to perform distance-based clustering to produce 9 sample clusters. Each relevant dimension of a cluster represents a gene that has similar expression levels in the member samples of the cluster, which is a potential signature of the sample type.

Leukemia: It consists of 38 bone marrow samples obtained from acute leukemia patients [14], 27 of them were diagnosed as ALL (acute lymphoblastic leukemia) and 11 as AML (acute myeloid leukemia). Each sample is described by the expression values of 7129 genes. The ALL samples can be further classified into two classes, one containing 19 B-cell ALL samples (B-ALL), and the other containing 8 T-cell ALL samples (T-ALL). We used HARP to perform distance-based clustering on the dataset to form two sample clusters, and compare the results with the ones presented in [14].

Yeast: The original dataset was published in [8]. It contains the expression levels of 6,218 yeast ORFs at 17 time points taken at 10 minute intervals, which cover nearly two full cell cycles. The dataset used here is the subset selected according to [26] that contains 2,884 genes. We preprocessed the data according to the method suggested in [7], and used HARP to perform pattern-based clustering to produce non-disjoint gene clusters using the two extensions. As in [7], we treated two genes as similar if they have complementary expression patterns in the corresponding subspace, i.e., the two genes constantly show opposite rise and fall patterns across the relevant dimensions. This is accomplished by having two copies of each gene in the dataset, one with the original expression values, and the other the negation of them. This results in two nearly identical copies of every cluster being formed. In the results reporting in the coming sections, all duplicated clusters and duplicated genes in a cluster are removed.

4.2 Results

The complete results can be found in the ancillary files. We summarize here some important findings.

Lymphoma: HARP was able to separate the samples of different types to different clusters with only a small number of errors. Some interesting clusters located at the top two levels of the dendrogram are listed in Table 2. We investigated the importance of dimension selection in the

Samples	No. of selected genes	A_1	A_2	A_3
6 RAT	2456	0.72	1.32	0.87
43 DLBCL, 2 NILNT	3515	0.96	1.25	1.02
10 ABB, 1 TCL	2734	0.80	1.32	1.00
9 FL, 2 GCB, 2 RBB	3104	0.85	1.38	1.00
11 CLL, 2 RBB	2614	0.82	1.27	0.97
16 DLBCL	3347	0.90	1.38	1.01
27 DLBCL, 2 NILNT	3610	0.96	1.32	1.00

Table 2: The distance ratios of some interesting clusters identified by HARP from the lymphoma data. Abbreviations: ABB=activated blood B, CLL=mantle cell lymphoma and chronic lymphocytic leukemia, DLBCL=diffuse large B-cell lymphoma, FL=follicular lymphoma, GCB=germinal centre B, RAT=resting/activated T, RBB=resting blood B, NILNT=NI. lymph node/tonsil, TCL=transformed cell lines.

formation of the clusters by calculating the distance ratios A_1 to A_3 defined as follows:

$$A_1(C_I) = \frac{\sum_{x_i \in C_I, v_j \in V_I} (x_{ij} - x_{Ij})^2 / d_I}{\sum_{x_i \in C_I, v_j \in V} (x_{ij} - x_{Ij})^2 / d} \tag{6}$$

$$A_2(C_I) = \frac{\sum_{x_i \in C_I, v_j \notin V_I} (x_{ij} - x_{Ij})^2 / (d - d_I)}{\sum_{x_i \in C_I, v_j \in V} (x_{ij} - x_{Ij})^2 / d} \tag{7}$$

$$A_3(C_I) = \frac{\sum_{x_i \notin C_I, v_j \in V_I} (x_{ij} - x_{Ij})^2 / d_I}{\sum_{x_i \notin C_I, v_j \in V} (x_{ij} - x_{Ij})^2 / d} \tag{8}$$

A_1 measures the increase in compactness of the cluster due to dimension selection, A_2 measures how irrelevant are the non-selected dimensions, and A_3 measures the increase in separation between the cluster members and other objects due to the selection. For a good cluster, A_1 should be smaller than one, A_2 should be greater than one, and A_3 should be larger than A_1. All clusters in Table 2 satisfy the three requirements, which means the selection of relevant dimensions makes the cluster members more distinguishable. For each cluster of samples, we also randomly selected 100,000 sets of relevant dimensions and calculated the corresponding distance ratios. All the resulting ratios are very close to one with standard deviations not more than 10^{-5}, which verify that the relevant dimensions selected by HARP are statistically unexpected and significantly better than random selections.

We then examined the biological meaning of the selected dimensions of the clusters. In Figure 2 of [4], some genes are highlighted as the signatures of some sample types or biological processes. The genes are divided into four regions: proliferation, germinal centre B, lymph node and T cell. For each cluster formed by HARP, we sorted all the genes in descending order according to their R values, and checked the ranks of the signature genes. It was found that the large DLBCL cluster

Algorithm	Best ARI
HARP	0.75
PROCLUS	0.64
KPrototype	0.63
CLARANS	0.61
Hierarchical	0.49
CAST	0.48

Table 3: The best ARI values achieved by various algorithms on the lymphoma data.

has many signature genes in the proliferation region receiving high ranks, which suggests that the expression values of the genes could potentially be used to identify DLBCL samples. Similarly, it was found that the resting/activated T samples have a distinctive expression pattern. The 6 samples form a clear cluster with many of the signature genes receiving very large R values. Activated blood B, FL and CLL samples formed three separate clusters consisting of few samples from other types. They all have large R values at the signature genes at the lymph node region due to the constantly low expression, but the three types of samples were successfully separated into different clusters according to the expression values of other relevant genes, in particular those in the germinal centre B region.

We also used the known sample types to evaluate the clustering accuracy. We used Adjusted Rand Index [28] as the performance metric, with the maximum value of one indicating a perfect clustering and zero indicating the clustering is no better than a random partitioning. We compared the best results of various projected and non-projected clustering algorithms, including a hierarchical method, the k-means method KProtype [16], the k-medoid method CLARANS [22], the CAST method [6] designed for clustering gene expression datasets, and the projected clustering method PROCLUS [1]. The results (shown in Table 3) suggest that the projected clustering methods have better performance in general, and HARP has the highest accuracy.

Leukemia: In [14], 50 informative genes that have very different expression patterns in the two classes are used to build a highly accurate classifier. This suggests that a very small number of relevant genes are enough to distinguish the two types of samples. We therefore initialized d_{min} to 50 in order to select a small set of highly relevant genes for each cluster. Notice that unlike setting the l parameter of PROCLUS and ORCLUS, initializing d_{min} to a certain value does not force HARP to select any specific number of genes for each cluster. HARP is free to select any number of genes *not less than* d_{min}. The setting simply suggests HARP to focus on the genes with larger R values. With this setting, HARP produced one cluster that contained only ALL samples and the other contained mainly AML samples with only 3 errors, which is a mild improvement over

Samples	No. of relevant genes	A_1	A_2	A_3
16 B-ALL, 8 T-ALL	112	0.40	1.01	2.97
11 AML, 3 B-ALL	59	0.35	1.00	2.81
8 T-ALL	151	0.24	1.01	2.07

Table 4: The distance ratios of the two final clusters and the pure T-ALL cluster identified by HARP from the leukemia data.

the clustering result presented in [14] (4 errors). The ALL and AML clusters identified by HARP have 112 and 59 selected genes respectively, both with average R values of 0.95, which indicate the extremely high distinguishing power of the genes. By examining the dendrogram, we also found that the 8 T-ALL samples formed its own cluster before merging with any B-ALL samples. The pure T-ALL cluster has 151 selected dimensions with average R value of 0.99, which are potential signature genes for distinguishing T-ALL from the other two types of samples.

The distance ratios A_1 to A_3 of the two final clusters and the T-ALL cluster are shown in Table 4. Comparing the ratios with those of the lymphoma clusters (Table 2), the A_1 ratios are much lower and the A_3 ratios are much higher. This indicates that dimension selection is more beneficial to the leukemia dataset by making the clusters more compact and more distant from each other. In contrast, the A_2 ratios are just slightly larger than one since only a small amount of dimensions are selected for each cluster, which means object distances in the non-selected subspace are not much different from those calculated in the input space.

Yeast: We used HARP to produce about 100 distinct clusters and compared them with the 100 biclusters discovered in [7]. Table 5 compares some statistics of the two sets of clusters. The H score of a cluster is the average squared residue score defined as follows:

$$H_I = \frac{\sum_{x_i \in C_I, v_j \in V_I} (x_{ij} - x_{Ij} - x_{iJ} + x_{IJ})^2}{N_I d_I}, \tag{9}$$

where x_{iJ} and x_{IJ} are the row average and block average respectively:

$$x_{iJ} = \frac{1}{d_I} \sum_{v_j \in V_I} x_{ij} \tag{10}$$

$$x_{IJ} = \frac{1}{N_I d_I} \sum_{x_i \in C_I, v_j \in V_I} x_{ij} \tag{11}$$

The lower is the H score, the more similar are the rise and fall patterns of the expression values of different objects. On average the clusters produced by HARP contain more genes but fewer time points. They also have a slightly better average squared residue score to size (number of genes multiplied by number of time points) ratio. Figure 3 shows the clusters with the best scores. According to the results, HARP was able to identify clusters with diverse sizes and dimensionalities. It also

Algorithm	Avg. no. of genes	Avg. no. of time points	Avg. H score	Avg. score to size ratio
Cheng and Church	167	12	204	0.10
HARP	243	10	203	0.08

Table 5: Comparison of the clusters identified by HARP and those reported Cheng and Church 2000 from the yeast data.

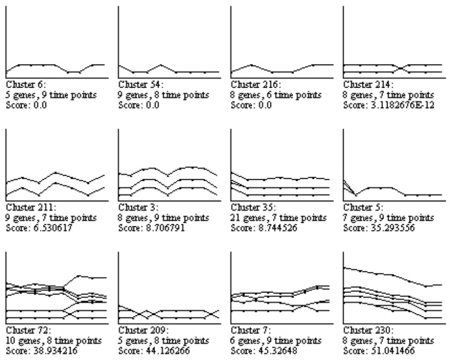


Figure 3: The clusters identified by HARP from the yeast data with the best mean squared residue scores. The genes in the same cluster are seen to have very similar expression patterns. Note that HARP is also able to cluster genes that constantly show opposite rise and fall patterns (see, for example, cluster 234).

successfully grouped together genes with similar expression patterns but in opposite directions. The average size of the clusters suggests that a significant number of genes were assigned to multiple clusters with matched signatures.

We evaluated the biological significance of the clusters by a phenotypic categorization of mRNAs that are regulated with the cell cycle (`http://yscdp.stanford.edu/yeast_cell_cycle/functional_categories.html`). Some clusters were found to contain a significant amount of genes from related categories. One such clusters is shown in Table 6, which contains many categorized genes in the late G1 phase, with functions ranging from budding, cell cycle regulation, nuclear segregation to DNA replication and repair.

It is interesting to see how similar are the clusters produced by HARP and those reported in [7]. For each cluster produced by HARP, we searched for a cluster reported in [7] with the highest Jaccard index. For every pair of clusters, we computed the Jaccard index by diviing the size of the intersection of the two sets of genes by the size of their union. An index value of one indicates two identical sets of genes, while an index value of zero indicates the two sets have no genes in common. The average, maximum and minimum Jaccard index so computed are 0.2102, 0.5872 and 0.0051

Category: genes
Budding, directional growth: YDR507C
Cell cycle regulators: YPL256C, YJL187C
Chromosome, nuclear segregation: YMR076C, YDL003W, YKL042W, YMR078C
DNA repair and recombination: YLR383W, YDR097C
DNA replication: YOR074C, YLR103C, YAR007C, YNL312W, YDL164C, YBR088C

Table 6: One of the clusters (cluster 53, no. of genes=22) identified by HARP from the yeast data that contains a significant amount of genes from related categories (all in late G1 phase).

respectively. The low index values suggest that the two sets of clusters are quite different, even they both have good average H scores. Although the two sets of results are not directly comparable due to the use of different parameter values (reflected by the difference in average cluster size and dimensionality), we believe the two methods do identify some clusters missed by the other, and that there are rooms for improvements in the topic of pattern-based clustering for gene expression data.

5. DISCUSSIONS

The results show that HARP is able to identify statistically and biologically meaningful clusters without relying on some user parameters whose proper values are hard to determine. It can thus be used to automatically identify some interesting clusters from a large number of datasets for later, more labor-intensive analysis.

The object assignment extension discovered some interesting non-disjoint clusters from the yeast dataset, but in general some important clusters could be missed if their structures are not captured by some disjoint clusters before object assignment. We propose two future extensions of HARP for identifying these clusters: to allow each cluster to be merged with multiple clusters, and to produce disjoint clusters on different small data samples, and then reassign other objects to the clusters. Both approaches allow the discovery of more projected structures.

The quality of the yeast clusters produced by HARP is comparable to those produced by the Cheng and Church algorithms, which were designed to optimize the pattern-based objective score. This suggests that non-projected clustering methods that assume distance-based object similarity can also be used in pattern-based clustering. Actually, in non-projected clustering, it can be easily proved that by standardizing a dataset such that each row has zero sum and unit sum of squares, the Euclidean distance between two objects in the transformed data is equal to $2 - 2r$, where r is the Pearson correlation between the objects in the original data [5]. This means the Euclidean distance between two objects in the transformed data reflects the dissimilarity between the rise and fall patterns of the objects in the original data. The pattern-based clustering problem is thus transformed to a distance-based clustering problem by the normalization process. The situation is more complicated in the projected case in that each cluster has its own set of relevant dimensions. As discussed in Section 3.6, normalization should be performed based on the projected values on such dimensions only. The trickiest thing is that the real relevant dimensions are unknown when normalization is performed. It becomes even more complicated when clusters are non-disjoint, at which a single projected value is subject to the normalization process of all the clusters that it is involved. We leave the more advanced methods of adaptive subspace normalization as a future work on the topic.

A well-known weakness of hierarchical clustering algorithms is the deterministic property: once an object is assigned to a cluster, it cannot be reassigned to another cluster. The object reassignment process performed at the end of clustering helps redistribute each object to the most similar cluster, but it is unable to correct wrong merges during the early stage of clustering. We have attempted to perform an object reassignment at the end of each threshold loosening step, but no significant accuracy improvements were observed, and the clustering process was severely prolonged. We will try to integrate the threshold loosening mechanism into other more efficient and non-deterministic clustering methods.

6. CONCLUSION

In this paper, we analyzed the major challenges of the projected clustering problem, and suggested some potential weaknesses of some existing projected clustering algorithms. Based on the analysis, we proposed a new projected clustering algorithm HARP that does not rely on user inputs in determining the relevant dimensions of clusters, which makes it easy to apply to applications where the correct values of the parameters are unknown. HARP makes use of the relevance index, histogram-based validation and dynamic threshold loosening to dynamically adjust the merging requirements of clusters according to the current clustering status. It also allows users to input some available domain knowledge, and it can be extended to perform pattern-based clustering and produce non-disjoint clusters by adaptive mean centering and post-clustering object assignment respectively. The experimental results on real microarray datasets show that HARP works well in situations where object similarity is based on either distance or expression pattern, and where disjoint or non-disjoint clusters are required. The clusters identified are both statistically and biologically meaningful. Future works include improving the speed performance of HARP and studying some advanced normalization techniques for projected pattern-based clustering.

7. ACKNOWLEDGEMENT

The research of DWC is supported by a grant from the Research Grant Council of Hong Kong. (Project number: HKU 7141/03E). KHC is supported in part by NIH grant K25 HG02378 from the National Human Genome Research Institute and NSF grant DBI-0135442.

8. REFERENCES

[1] C. C. Aggarwal, C. Procopiuc, J. L. Wolf, P. S. Yu, and J. S. Park. Fast algorithms for projected clustering. In *ACM SIGMOD International Conference on Management of Data*, 1999.

[2] C. C. Aggarwal and P. S. Yu. Finding generalized projected clusters in high dimensional spaces. In *ACM SIGMOD International Conference on Management of Data*, 2000.

[3] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *ACM SIGMOD International Conference on Management of Data*, 1998.

[4] A. A. Alizadeh, M. B. Eisen, R. E. Davis, C. Ma, I. S. Lossos, A. Rosenwald, J. C. Boldrick, H. Sabet, T. Tran, X. Yu, J. I. Powell, L. Yang, G. E. Marti, T. Moore, J. Hudson, L. Lu, D. B. Lewis, R. Tibshirani, G. Sherlock, W. C. Chan, T. C. Greiner, D. D.Weisenburger, J. O. Armitage, R. Warnke, R. Levy, W. Wilson, M. R. Grever, J. C. Byrd, D. Botstein, P. O. Brown, and L. M. Staudt. Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, 403:503–511, 2000.

[5] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Natl. Acad. Sci. USA*, 96:6745–6750, 1999.

[6] A. Ben-Dor and Z. Yakhini. Clustering gene expression patterns. In *Proceedings of the Annual International Conference on Computational Molecular Biology*, 1999.

[7] Y. Cheng and G. M. Church. Biclustering of expression data. In *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology*, 2000.

[8] R. J. Cho, M. J. Campbell, E. A. Winzeler, L. Steinmetz, A. Conway, L. Wodicka, T. G. Wolfsberg, A. E. Gabrielian, D. Landsman, D. J. Lockhart, and R. W. Davis. A genome-wide transcriptional analysis of the mitotic cell cycle. *Molecular Cell*, 2:65–73, 1998.

[9] F. De Smet, J. Mathys, K. Marchal, G. Thijs, B. De Moor, and Y. Moreau. Adaptive quality-based clustering of gene expression profiles. *Bioinformatics*, 18(5):735–746, 2002.

[10] D. Dembele and P. Kastner. Fuzzy C-means method for clustering microarray data. *BioInformatics*, 19(8):973–980, 2003.

[11] 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*, 95:14863–14868, 1998.

[12] D. Eppstein. Fast hierarchical clustering and other applications of dynamic closest pairs. In *SODA: ACM-SIAM Symposium on Discrete Algorithms*, 1998.

[13] A. P. Gasch, P. T. Spellman, C. M. Kao, O. Carmel-Harel, M. B. Eisen, G. Storz, D. Botstein, and P. O. Brown. Genomic expression programs in the response of yeast cells to environmental changes. *Molecular Biology of the Cell*, 11:4241–4257, 2000.

[14] 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, 1999.

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

[16] Z. Huang. Clustering large data sets with mixed numeric and categorical values. In *The First Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 1997.

[17] V. R. Iyer, M. B. Eisen, D. T. Ross, G. Schuler, T. Moore, J. C. F. Lee, J. M. Trent, L. M. Staudt, J. H. Jr., M. S. Boguski, D. Lashkari, D. Shalon, D. Botstein, and P. O. Brown. The transcriptional program in the response of human fibroblasts to serum. *Science*, 283:83–87, 1999.

[18] Y. Kluger, R. Basri, J. T. Chang, and M. Gerstein. Spectral biclustering of microarray cancer data: Co-clustering genes and conditions. *Genome Research*, 13(4):703–716, 2003.

[19] L. Lazzeroni and A. Owen. Plaid models for gene expression data. *Statistica Sinica*, 12:61–86, 2002.

[20] 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, 2001.

[21] D. Matei, T. G. Graeber, R. L. Baldwin, B. Y. Karlan, J. Rao, and D. D. Chang. Gene expression in epithelial ovarian carcinoma. *Oncogene*, 21:6289–6298, 2002.

[22] R. T. Ng and J. Han. Efficient and effective clustering methods for spatial data mining. In *20th International Conference on Very Large Data Bases, September 12–15, 1994, Santiago, Chile proceedings*, 1994.

[23] S. L. Pomeroy, P. Tamayo, M. Gaasenbeek, L. M. Sturla, M. Angelo, M. E. McLaughlin, J. Y. H. Kim, L. C. Goumnerovak, P. M. Black, C. Lau, J. C. Allen, D. Zagzag, J. M. Olson, T. Curran, C. Wetmore, J. A. Biegel, T. Poggio, S. Mukherjee, R. Rifkin, A. Califano, G. Stolovitzky, D. N. Louis, J. P. Mesirov, E. S. Lander, and T. R. Golub. Prediction of central nervous system embryonal tumour outcome based on gene expression. *Nature*, 415:436–442, 2002.

[24] C. M. Procopiuc, M. Jones, P. K. Agarwal, and T. M. Murali. A monte carlo algorithm for fast projective clustering. In *ACM SIGMOD International Conference on Management of Data*, 2002.

[25] J. R. Quinlan. *C4.5 Programs for Machine Learning*. Morgan Kaufmann, 1993.

[26] S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church. Systematic determination of genetic network architecture. *Nature Genetics*, 22:281–285, 1999.

[27] J. Yang, H. Wang, W. Wang, and P. Yu. Enhanced biclustering on expression data. In *Proceedings of the IEEE Third Symposium on Bioinformatics and Bioengineering*, 2003.

[28] K. Yeung and W. Ruzzo. An empirical study on principal component analysis for clustering gene expression data. *Bioinformatics*, 17(9):763–774, 2001.

[29] K. Y. L. Yip. HARP: A practical projected clustering algorithm for mining gene expression data. Master's thesis, The University of Hong Kong, Pokfulam Road, Hong Kong, 2004. `http://www.csis.hku.hk/~ylyip/papers/thesis.pdf`.