Gene clustering methods for time series microarray data

Laney Kuenzel

Biochemistry 218

June 6, 2010

1 Introduction

The development of advanced microarray technology over the past two decades constitutes a revolution in genomics. Today, microarrays can measure expression levels for thousands of genes simultaneously, yielding data with the potential to shed light on many areas of biology and medicine, from the cell cycle to physiological development to disease mechanisms. The main challenge for researchers lies in drawing meaningful conclusions from the vast quantities of microarray data that they collect.

Clustering, one of the key steps in making sense of microarray data, divides the set of genes being studied into smaller sets of genes with similar expression patterns, where the definition of "similar" varies from algorithm to algorithm. According to the widely-accepted "guilt by association" principle (the validity of which is supported by [45]), genes that are co-expressed are very often co-regulated as well. Therefore, by revealing which genes are co-expressed, clustering assists researchers in discovering and understanding the regulatory mechanisms of the cells in question.

Microarray experiments can be divided into two main types: static and time series. In static experiments, gene expression measurements are taken one time each from a number of samples. For example, in a common type of static microarray experiment, researchers studying the mechanism of a particular disease measure and compare gene expression levels in tissue samples taken from individuals with and without the disease [41].

In time series (or temporal) experiments, on the other hand, expression levels are measured in a single sample at a number of points in time. Most of the applications of time series microarray experiments can be placed into one of four broad categories. The first is discovering the dynamics behind various biological systems, such as the cell cycle and the circadian clock [1, 3]. The second category is development; by collecting and analyzing time series data during a particular developmental process, researchers can learn about the genes controlling that process. Interesting examples of processes that have been studied in this way include nervous system development and stem cell differentiation [1, 41]. Third, temporal microarray

experiments can shed light on disease progression by revealing the genetic changes underlying observable symptoms [1, 3]. Researchers have applied microarray technology to study diseases such as Alzheimer's [18], HIV [37], and cancer [44]. Fourth and finally, researchers can use time series experiments to determine genetic responses to various conditions of interest, such as gene knockouts, stress conditions, and drug administration [17, 41].

Temporal expression data clearly has the potential to generate a great deal of biological knowledge. For researchers hoping to start with raw time series microarray data and reach the type of useful results described above, data analysis constitutes the most challenging step. In particular, there is little to no consensus in the literature about the best method for clustering time series microarray data despite the fact that hundreds of algorithms have been developed for the task. Before going into the details of such algorithms, we will briefly describe the general categories of clustering procedures.

2 Clustering Approaches

The goal of clustering is to group similar data points together. Accordingly, any clustering algorithm can be characterized by the notion of similarity that it employs. As described in [50], clustering algorithms can be divided into two main types: discriminative and generative. Discriminative algorithms define a pairwise similarity function and then apply that function to cluster similar data points together. Generative algorithms, by contrast, assume that the data is generated by a finite set of models. Such algorithms use the data to learn the parameters for those underlying models and then cluster points generated by the same model together.

Below, we review the application of both types of algorithms to the analysis of time series microarray data and consider the advantages and disadvantages of each approach.

3 Discriminative Algorithms

In the specific case of clustering temporal microarray data, the objects being grouped together are the expression profiles for individual genes. At the root of any discriminative algorithm, then, is a function measuring similarity between two expression profiles. These similarity functions range from straightforward pointwise distance metrics to more complex functions based on general features extracted from the profile.

3.1 Pointwise Similarity

Most traditional clustering algorithms such as hierarchical, k-means, and self-organizing maps fall into this category. In essence, these algorithms treat expression profiles with n time points as n-dimensional vectors and apply distance or correlation functions to those vectors in order to quantify the similarity between two profiles. Euclidean distance, Manhattan distance, and the Pearson correlation coefficient are among the most common functions chosen [10].

We can subdivide pointwise discriminative algorithms even further based on how they apply the defined pointwise similarity function. Some algorithms, including most hierarchical methods, work in an agglomerative way by building clusters up from individual data points and smaller clusters [7]. Others, most notably k-means and self-organizing maps, work divisively by breaking the set of all data points down into smaller clusters [6].

Pointwise discriminative algorithms have been successfully applied to cluster static microarray data [7]. Furthermore,

they are generally simple to understand and implement. As a result, these algorithms were the ones chosen in the earliest time series microarray experiments [12].

Unfortunately, despite their relative simplicity, pointwise discriminative algorithms are not appropriate for the analysis of temporal microarray data. The validity of these algorithms relies on the assumption that the expression measurements taken for a given gene are independently and identically distributed [31]. This assumption is reasonable for static data taken from different samples; however, it certainly does not hold for temporal data, in which nearby time points are strongly correlated with one another [3].

To phrase the same fundamental issue in another way, pointwise discriminative algorithms ignore the temporal content of time series data. The order of expression measurements is irrelevant for the purpose of these algorithms. Clearly, the analysis of time series expression data calls for methods that make effective use of temporal information rather than disregarding it.

3.2 Feature-Based Similarity

Feature-based discriminative algorithms compare two genes not based on the raw expression data—as is the case with pointwise algorithms—but rather based on a set of features extracted from that data. Essentially, these methods first transform each gene expression vector into a feature vector to which they then apply the traditional pointwise clustering methods described above.

Ideally, the feature vector should encapsulate the most important aspects of an expression profile. The challenge in designing a feature-based algorithm, then, is choosing which features to extract. In [11], Di Camillo et al. proposed a method in which each of a gene's expression measurements is replaced by either -1, $+1$, or 0 depending on whether the gene is under-expressed, over-expressed, or not differentially expressed relative to its baseline expression level.

Phang et al. [34] took a different approach, choosing to transform each gene expression vector into a trajectory vector in which each term indicates the direction of change (either increase, decrease, or flat) between two consecutive time points in the original vector. Other researchers have since proposed extensions to their algorithm. For instance, Kim and Kim [23] extracted not only a sequence of first-order differences but also a sequence of second-order differences (convex, concave, or no change) from the original gene expression vectors. They then combined the two sequences into a single feature vector.

Phan et al. [33] proposed a method combining aspects of both Di Camillo's and Phang's approaches. In their procedure, the feature vector contains expression levels relative to a baseline value (up-modulated, no change, or down-modulated) as well as differences in pairs of successive expression measurements (steep rise, moderate rise, no change, moderate fall, or steep fall).

StepMiner [39] and SlopeMiner [29] provide two final examples of feature-based algorithms. Both methods rely on the assumption that transitions in expression level are the defining characteristics of an expression profile. In the StepMiner algorithm, each expression vector is reduced to a simple binary pattern (up, down, up-down, down-up, or none) and, depending on the assigned pattern, the one or two times corresponding to the transitions. SlopeMiner extends StepMiner to allow for the identification of gradual transitions in expression level rather than just sharp ones.

There are a number of compelling advantages to feature-based discriminative algorithms. For one, feature-based algo-

rithms are often faster than other clustering methods because the actual clustering step is performed not on the original expression vectors but on the simpler feature vectors [19]. Moreover, because the feature extraction step usually reduces noise present in the raw microarray data, feature-based methods tend to be more robust than algorithms that operate directly on the original untransformed data [42]. Most importantly, feature-based approaches are flexible; they allow researchers to reduce a complex expression profile down to those characteristics that they consider essential [1]. As a result, genes are compared based on the most important aspects of their profiles, and meaningful clusters are generated.

In addition to their many advantages, feature-based algorithms also have significant flaws. For one, feature extraction necessarily involves the loss of information. Even if features are carefully chosen, there is no way to ensure that feature vectors will capture all of the important patterns present in the data. Furthermore, as discussed in [42], with the development of a feature-based algorithm comes the danger of introducing bias into the data analysis. The main goal of clustering gene expression data is to discover previously unknown groups of co-expressed genes. By selecting features based on patterns that they expect to observe in the expression profiles, researchers may prevent the discovery of unexpected patterns and similarities.

One simple way to reduce the chance of losing important information or introducing bias is to cluster the data using a variety of different feature sets. This solution is certainly feasible since, as mentioned above, feature-based algorithms are relatively fast. Furthermore, the underlying code could be written so that updating the set of extracted features would require minimal changes. By examining the clusters generated for several choices of feature sets, researchers would be more likely to discover novel groupings of co-expressed genes.

The ideal solution would be to develop an algorithm that automatically selects the optimal features from a large set of previously defined candidate features. In the machine learning literature, the various approaches to automatic feature selection are categorized as either filters or wrappers [43]. In the context of microarray clustering methods, filters would use annotated data (perhaps from the Gene Ontology database) to choose the feature subset that most successfully clusters together groups of genes known to be co-regulated. Wrappers, on the other hand, would apply each possible feature subset to the data in question and use the one that generated the most internally valid clusters. Because the latter task is extremely computationally demanding, filters are the more realistic option in the short term [47].

3.3 Shape-Based Similarity

A third and final class of discriminative algorithms use the notion of shape to define their similarity functions. For instance, Qian et al. [35] described an algorithm that identifies profiles with a given shape as similar, regardless of whether that shape is time-shifted or inverted. Their procedure, based on the Smith-Waterman algorithm for local sequence alignment, assigns to each pair of expression profiles a score and a relationship: simultaneous, time-delayed, inverted, or inverted time-delayed. The score (weighted by the relationship, if desired) can then be taken as a measure of similarity and used for clustering the genes.

Several researchers have proposed improvements to the technique described above. For example, Balasubramaniyan et al. [2] developed a heuristic algorithm analogous to BLAST (Basic Local Alignment

Search Tool) to speed up the process of finding the maximal local alignment. He and Zeng [20] incorporated concepts from feature extraction into Qian's procedure; they proposed that prior to alignment, gene expression vectors should be transformed into "change trend" vectors containing the direction of change in gene expression levels for successive time points.

The major advantage of these shape-based algorithms lies in their ability to identify as similar two expression profiles that are shifted, inverted, or both. From a biological perspective, the shifted relationship corresponds to one gene regulating another or to one gene having a time delay in its response to the same transcription factor. The inverted correlation suggests that the same regulatory mechanism activates one gene and inhibits the other. By detecting these important types of similarity that other methods miss, these algorithms have the potential to uncover new connections among genes. Indeed, Qian et al. [35] reported that their algorithm identified several novel inverted and time-delayed relationships among yeast cell cycle genes, suggesting that their shape-based approach has promise for clustering other types of temporal microarray data as well.

There are a two simple ways in which the shape-based algorithms could be improved. For one, allowing penalized gaps in the alignment is mentioned in [35] but not implemented in any of the three procedures. The inclusion of gaps would reduce problems caused by non-uniform sampling of time points. Second, the algorithms could more effectively capture profile shape by utilizing techniques such as dynamic time warping, which tolerates local stretching or compression of one profile relative to the other [22], and uniform scaling, which allows for one entire profile to be stretched relative to the other [15].

The main drawback of these shape-based algorithms is that they are very computationally demanding; the slow process of finding the best local sequence alignment must be performed many times to create the clusters. The challenge for algorithm developers is to design good heuristic approaches that make the search faster without compromising too much accuracy, as in [2].

4 Generative Algorithms

The fundamental idea motivating generative algorithms is that a finite set of processes or models generate the gene expression data. Rather than directly measuring the similarity between pairs of expression profiles as do discriminative methods, generative algorithms use the data to determine the optimal parameters for the underlying models and then identify as similar any profiles generated by the same model.

4.1 Template-Based

In template-based (also called template-matching) algorithms, each gene expression vector is placed into the cluster corresponding to the particular "template," or candidate profile, that describes it most closely. Although template-based approaches have much in common with the feature-based methods described in Section 3.2, we classify them as generative algorithms because they work not by measuring similarity between pairs of expression profiles but by assigning each expression profile to the candidate profile it matches best. Template-matching algorithms differ from one another primarily in the preprocessing steps they apply to the data and in their method for choosing the candidate profiles.

One of the early template-based algorithms, which is also one of the most frequently used, was presented by Peddada et al. in [32]. The first step of their proce-

dure is the definition of candidate inequality profiles, such as monotone decreasing or cyclical, by the experimenter. The second step utilizes statistical techniques to match each expression profile to one of the candidates or, if a unique significant match is not found, to none of them.

Liu et al. [26] described a modified version of Peddada's algorithm that runs faster and also provides a measure of cluster significance for the researcher. Yi et al. [49] updated Peddada's algorithm to operate not on the original gene expression vectors but on discretized rank vectors instead. They found that this adjustment led to improved performance on data with high variability.

If the experimenter knows prior to the data analysis step which expression patterns he or she hopes to identify, then approaches like Peddada's are appropriate for creating gene clusters. However, in many cases, the goal of clustering microarray data is to identify new relationships among genes based on unanticipated patterns and similarities. In this case, it does not make sense for an algorithm to use candidate expression profiles pre-specified by the researcher.

To address this issue, Moller-Levet et al. [30] presented a template-matching algorithm that does not require researchers to enter candidate profiles. In the first step of their method, each gene expression vector is transformed to a "pattern vector" indicating the sign of change (increase or decrease) for each pair of consecutive expression measurements. The unique aspect of their procedure is that it includes every possible pattern vector as a template profile. Each gene is assigned to the cluster defined by its exact pattern vector.

With this algorithm, researchers no longer have to choose their own candidate profiles. However, as time series get longer, the number of template profiles and therefore the number of clusters becomes large compared to the number of genes. For example, suppose that Moller-Levet's algorithm were applied to time series data on 2000 different genes with 12 time points. There would be $2^{12-1} = 2048$ clusters, so the expected size of a cluster would be less than one. Such small clusters are certainly not conducive to the identification of groups of co-expressed genes.

Clearly, both approaches discussed so far (using experimenter-defined candidates and using all possible profiles as candidates) lead to significant problems. Ernst et al. [13] avoided both types of issues by developing an algorithm that selects a reasonably-sized yet still representative subset of all possible expression profiles to use as templates. Because finding the best such subset of a given size (where best is formally defined as containing the profiles most distinct from one another in a pairwise sense) is an NP-hard problem, Ernst et al. proposed a greedy algorithm guaranteed to find a good, though not necessarily optimal, subset. The main drawback to this method is that, as described in [13], the subset selection algorithm is prohibitively slow given a large space of possible profiles, which can result from either a large number of time points or a high level of detail in the extracted pattern vectors.

Overall, template-based methods have compelling strong points. Because they deal with pattern vectors rather than the raw data, they are robust to noise [38]. They work particularly well for short time series, which is important because, as noted in [13], over eighty percent of all time series in the Stanford Microarray Database contain fewer than nine time points. As long as the use of template-based algorithms is restricted to short time series, the slowness of the methods should not pose a problem.

The primary disadvantage of template-based approaches is the same as that for feature-based methods: the potential loss

of information resulting from the transformation of the original expression vector into a pattern vector. For example, none of the template-based algorithms described here retain information about second-order change in expression measurements—a feature that could be useful in identifying co-expressed genes. Addressing this issue is not as simple as incorporating second-order information into the pattern vector, since such an increase in the complexity of the pattern vector would introduce many more possible templates and, as a consequence, lead to significantly higher running time of the algorithms.

The fundamental issue here is that a good deal of information must be lost in order to keep the space of possible templates sufficiently small. To exemplify this concept, we consider a final template-matching approach proposed by Liu et al. [25]. Their method is unique in the preprocessing procedure it applies to the raw expression data. For a given profile, rather than extracting information about the direction of change between consecutive expression measurements, their algorithm extracts a qualitative description (such as "linear up," "concave up," or "convex down") of the quadratic regression that best fits that profile. They have one template, and consequently one cluster, corresponding to each possible such description. The main strength of this approach is that it treats time as a continuous variable, making it a good choice for time series with non-uniform sampling. Unlike the methods previously discussed, this algorithm successfully takes into account the second-order nature of the profiles; however, it does so at the cost of losing many other details, such as local patterns in the profiles.

The above example illustrates how any template-matching algorithm suffers from the loss of potentially important information. One promising approach to addressing this problem is to combine template-based methods with other types of clustering algorithms. For instance, temporal microarray data for thousands of genes could initially be clustered by a template-based procedure. The genes in a single resulting cluster could then be further clustered using a different algorithm which retains more information, like one of the shape-based procedures discussed in Section 3.3. This combined approach would first group the genes based on very general shared patterns and then make further distinctions within any individual group based on the more complex aspects of the expression profiles. As an added bonus, the slowness of the shape-based algorithms would not be an issue because they would only need to run on a small subset of the overall set of genes.

4.2 Model-Based

Much more so than the template-matching clustering approaches discussed above, model-based methods serve as quintessential examples of generative algorithms. Associated with each model-based algorithm is a form for the parametric models assumed to generate the data. The algorithms apply statistical techniques, often some sort of expectation-maximization algorithm, to obtain maximum likelihood estimates for the parameters of the models [24]. Each model represents a single cluster, and a given gene is placed into the cluster corresponding to the model most likely to have generated its observed expression profile.

Not surprisingly, the defining feature of a model-based clustering algorithm is its choice of underlying models. Yeung et al. [48], for example, developed a method using multivariate normal distributions as models for the gene expression data. Their algorithm consists of two steps: the application of the Bayesian information criterion to

determine the appropriate number of clusters, and the use of a standard expectation-maximization algorithm to determine the parameters of each of the normal distributions. Overall, their approach is not particularly effective for clustering time series microarray data because, as noted in [31], it completely disregards the temporal nature of the expression profiles.

In [4], Bar-Joseph et al. presented a different model-based algorithm more appropriate for application to temporal data. Their method models expression profiles using splines, which are essentially continuous piecewise polynomial functions. Unlike many of the methods previously discussed, this algorithm retains and utilizes information about the duration of time between each sample. Consequently, it is particularly appropriate for clustering non-uniformly sampled temporal data. One major drawback of Bar-Joseph's method is its requirement of the number of clusters as input. Luan and Li [27] addressed this problem by developing a similar algorithm based on spline models which automatically determines the optimal number of clusters using the Bayesian information criterion. In [28], Ma et al. described a way to significantly increase the speed of the spline algorithms by making the maximization step of the expectation-maximization algorithm less computationally expensive.

Overall, while splines are good choices of models in that they represent expression profiles as continuous, they also have several serious downsides. For one, an algorithm using spline representations must pre-specify the number and length of the "pieces" making up the splines [27]. These specifications must either be made arbitrarily, or else significant time must be spent trying out different combinations of specifications to determine which give the best results. Unfortunately, both of these two options are problematic. Another issue with spline representations is that they do not work well for short time series, since accurately fitting the spline model requires a minimum number of about four time points in each of the several spline segments [5,25].

In a much-cited paper, Ramoni et al. [36] proposed an alternative type of model-based algorithm for clustering time series microarray data. As models, their method uses first-order autoregressive equations, in which each expression measurement is a linear function of the previous measurement. The rationale behind their choice of model is that autoregressive equations successfully capture the ways in which expression measurements depend on one another. To determine the most likely set of autoregressive models generating the data, their algorithm utilizes an agglomerative Bayesian clustering approach. More specifically, it starts by creating one cluster per gene and then merges two clusters whenever doing so increases the posterior probability of the set of models given the data. In [46], Wu et al. improved upon Ramoni's method by replacing the agglomerative search procedure with a less computationally expensive and more theoretically sound expectation-maximization algorithm.

Autoregressive equations are apt models for time series data because they account for the dependency of expression measurements. However, unlike spline representations, autoregressive models disregard the amount of time between samples, decreasing their effectiveness for non-uniformly sampled time series [31]. Furthermore, by assuming that time series data are stationary (that is, that the mean and variance stay constant over time [9]), autoregressive models miss potentially important time trends in the data [27].

A final group of model-based algorithms make use of hidden Markov models (HMMs). Like autoregressive models, HMMs account for the dependencies of gene

expression levels. Schliep et al. [40] proposed an approach in which each HMM state has its own Gaussian emission probability distribution over possible expression measurements. Their algorithm starts with a number of HMMs describing "prototypical" expression patterns and then applies an expectation-maximization algorithm to find the maximum likelihood parameters for the HMMs. Their method dynamically adjusts the number of HMM clusters by deleting those with too few profiles and splitting those with too many. Another interesting feature of their approach is the inclusion of what they call a "noise cluster," essentially a very general HMM that can generate any possible profile, in case the initial HMMs do not successfully represent some of the observed profiles.

Ji et al. [21] developed another algorithm for clustering time series data using HMMs. Their method differs from that of Schliep in two main ways. For one, their procedure starts by converting each expression vector to a "pattern vector" consisting of symbols for up, down, and no change like those described in Section 4.1. This transformation leads to a loss of information, but it also simplifies the HMMs used. Rather than requiring a continuous emission probability to be specified for each HMM state as does Schliep's algorithm, Ji's method only requires three emission probabilities for each state.

The second major difference between the two approaches is the method of selecting the number of clusters. While the determination of appropriate cluster number is built into Schliep's algorithm, Ji et al. choose the number of clusters through what boils down to a trial-and-error procedure; they run their algorithm for different possible cluster numbers and choose the one which produces the most internally consistent clusters based on the "figure of merit" metric. In this respect, Ji's approach is at a disadvantage because it requires an already computationally intensive algorithm to be run multiple times.

A general issue with approaches based on HMMs is that they, like autoregressive models, disregard information about how samples are distributed in time and therefore may not be effective for use with non-uniformly sampled data [31].

Overall, there are many convincing arguments for using model-based algorithms to cluster time series microarray data. For one, these algorithms are based on a sound theoretical foundation. Furthermore, clusters have clear interpretations; experimenters can easily determine the similarities among the genes in a given cluster by examining the model corresponding to that cluster. As noted in [48], the statistical framework behind model-based methods suggests a natural and sound way to choose the correct number of clusters: by using statistical criteria such as the Bayesian information criterion. Perhaps the most compelling advantage of model-based methods is that they not only create clusters but also provide experimenters with a statistical measure of how well each gene fits into its cluster and, if desired, into any other cluster [14]. Such statistical measures enable "soft clustering," whereby each gene is assigned some degree of membership (typically varying between zero and one) in each cluster rather than being put definitively into a single cluster [16]. Soft clustering algorithms not only provide experimenters with more potentially useful information about the relationships among genes, but also tend to be more robust to noise than so-called "hard clustering" algorithms [16].

Unfortunately, like all of the other types of clustering methods discussed up to this point, model-based algorithms have a few serious downsides. For one, the choice of model introduces a great deal of bias

into the clustering results and restricts the type of similarities that can be identified among genes. For each of several different choices of models (including splines, autoregressive equations, and HMMs), there are compelling arguments as to why that particular model is suitable for temporal microarray data. However, there is no definitive way to show in general that a given model describes time series gene expression data more accurately than another. As a result, it is difficult for experimenters to know which models to select to analyze their data. As discussed in Section 3.2 in the context of feature-based algorithms, one possible way to deal with this problem is to try clustering the data using several different models and see if any of them identify interesting new relationships that merit additional investigation. In Section 5, we discuss this possibility further.

A more concrete problem with model-based algorithms is that the search for the optimal model parameters can be very computationally demanding, particularly for models like HMMs with many parameters. Fortunately, a number of fast yet reasonably accurate stochastic versions of the expectation-maximization algorithm have been proposed in the general cluster analysis literature [8, 14], and these modifications could be used to increase the speed of any model-based gene expression clustering method if desired.

5 Combined Methods

One clear conclusion that can be drawn from the preceding review is that, for the purpose of clustering time series microarray data, no one method is definitively better than all others. Each algorithm relies on some assumption of what makes expression profiles similar and is therefore limited by that assumption in terms of the similarities it will identify and the clusters it will form.

A naive approach to avoid being limited by a single assumption is to apply a number of different clustering algorithms to the data. Since the overall purpose of clustering expression data is to discover new and unexpected groups of co-expressed genes, it seems that trying out more clustering algorithms can only improve the usefulness of the results [6]. This reasoning is generally valid, but it has a few flaws. First, the clustering algorithms discussed here require a significant amount of time and computational resources, so running several or many of these algorithms is often not a feasible option.

Besides this practical consideration, there is a more fundamental flaw in the approach of applying a large number of clustering algorithms: it has the potential to overwhelm the experimenter with excessive information. Recall that the broad goal of clustering (and, for that matter, data analysis in general) is to get from huge amounts of raw data that a human brain cannot possibly process to a manageable summary of the important patterns and relationships in that data. This goal will not be achieved if the experimenter has to consider and synthesize too many different sets of clusters.

In light of these considerations, there is a need for a way to run various clustering algorithms on the same data set—thereby taking advantage of all of their strengths and minimizing the total amount of information lost—yet still present the experimenter with a clear and manageable summary of the results. We propose the development of a software tool for this purpose. This tool would take raw gene expression data as input from the user and run several diverse clustering algorithms with complementary strengths on that data. In general, a good set of algorithms should include a feature-based generative algorithm, a shape-based generative algorithm, a model-based algorithm that represents

the profiles as continuous functions (such as splines), and a model-based algorithm that accounts for the dependencies in the data (such as autoregressive models or HMMs). Of course, there are many possible ways to choose which particular algorithms the tool would use, and making the best choices would require a thorough evaluation of the algorithms in terms of both running time and quality of results. One possibility is that the tool would utilize characteristics of the data itself—such as whether it is uniformly sampled and whether there are many or few time points—to choose the best set of algorithms to run.

The most important feature of this software tool would be a method of synthesizing, summarizing, and visually displaying the results of the various algorithms. One straightforward way to accomplish this synthesis step would be to perform a simple hierarchical meta-clustering of the genes, whereby the similarity of two genes would be defined by how many times they were clustered together by the various algorithms used. The user could select different tabs that would show him or her the meta-clusters as well as the clusters produced by any of the individual algorithms. Overall, this tool would give experimenters a simple way to benefit from the strengths of several different clustering approaches, making them more likely to obtain useful results.

6 Conclusion

In the past decade, hundreds of clustering algorithms have been developed particularly for application to time series microarray data. In one sense, the abundance of methods is positive for experimenters since it gives them many good options in selecting an algorithm. On the other hand, the existence of so many different clustering procedures makes choosing the appropriate one difficult, especially given that each algorithm has its own set of often subtle assumptions and works best for data with particular characteristics. Hopefully, reviews such as this one will provide researchers with information that will allow them to choose the algorithm most effective for their data, ultimately leading them to uncover new patterns of co-expression among genes and make important biological discoveries.

References

[1] I. P. Androulakis, E. Yang, and R. R. Almon. Analysis of time-series gene expression data: Methods, challenges, and opportunities. *Annu Rev Biomed Eng*, 9:205–228, 2007.

[2] R. Balasubramaniyan, E. Hullermeier, N. Weskamp, and J. Kamper. Clustering of gene expression data using a local shape-based similarity measure. *Bioinformatics*, 21:1069–1077, 2005.

[3] Z. Bar-Joseph. Analyzing time series gene expression data. *Bioinformatics*, 20(16):2493–2503, 2004.

[4] Z. Bar-Joseph, G. Gerber, D. K. Gifford, T. S. Jaakkola, and I. Simon. A new approach to analyzing gene expression time series data. In *RECOMB '02: Proceedings of the sixth annual international conference on Computational biology*, pages 39–48, New York, NY, USA, 2002. ACM.

[5] Z. Bar-Joseph, G. K. Gerber, D. K. Gifford, T. S. Jaakkola, and I. Simon. Continuous representations of time-series gene expression data. *J Comput Biol*, 10:341–356, 2003.

[6] A. Bergkvist, V. Rusnakova, R. Sindelka, J. M. Garda, B. Sjogreen, D. Lindh, A. Forootan, and M. Kubista. Gene expression profiling–Clusters of possibilities. *Methods*, 50:323–335, 2010.

[7] P. Boutros and A. Okey. Unsupervised pattern recognitions: An introduction to the whys and wherefores of clustering microarray data. *Brief Bioinform*, 6(4):331–43, 2005.

[8] G. Celeux and G. Govaert. A classification EM algorithm for clustering and two stochastic versions. *Comput Stat Data Anal*, 14(3):315–332, 1992.

[9] C. Chatfield. *The analysis of time series: An introduction*. CRC Pres LLC, Boca Raton, FL, 2004.

[10] P. D'Haeseleer. How does gene expression clustering work? *Nature Biotech*, 23(12):1499–1501, 2005.

[11] B. Di Camillo, F. Sanchez-Cabo, G. Toffolo, S. K. Nair, Z. Trajanoski, and C. Cobelli. A quantization method based on threshold optimization for microarray short time series. *BMC Bioinformatics*, 6 Suppl 4:S11, 2005.

[12] 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.

[13] J. Ernst, G. J. Nau, and Z. Bar-Joseph. Clustering short time series gene expression data. *Bioinformatics*, 21 Suppl 1:i159–168, 2005.

[14] C. Fraley and A. E. Raftery. How many clusters? Which clustering method? Answers via model-based cluster analysis. *The Computer Journal*, 41:578–588, 1998.

[15] A. W.-c. Fu, E. Keogh, L. Y. H. Lau, and C. A. Ratanamahatana. Scaling and time warping in time series querying. In *VLDB '05: Proceedings of the 31st International Conference on Very Large Databases*, pages 649–660. VLDB Endowment, 2005.

[16] M. E. Futschik and B. Carlisle. Noise-robust soft clustering of gene expression time-course data. *J Bioinform Comput Biol*, 3:965–988, 2005.

[17] 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. *Mol Biol Cell*, 11(12):4241–4257, 2000.

[18] S. D. Ginsberg, S. E. Hemby, S. E. Lee, V. M. Lee, and J. H. Eberwine. Expression profiling of transcripts in Alzheimer's disease tangle-bearing CA1 neurons. *Ann Neurol*, 48:77–87, 2000.

[19] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *J Mach Learn Res*, 3:1157–1182, 2003.

[20] F. He and A. P. Zeng. In search of functional association from time-series microarray data based on the change trend and level of gene expression. *BMC Bioinformatics*, 7:69, 2006.

[21] X. Ji, J. Li-Ling, and Z. Sun. Mining gene expression data using a novel approach based on hidden Markov models. *FEBS Lett*, 542:125–131, 2003.

[22] E. J. Keogh and M. J. Pazzani. Derivative dynamic time warping. In *First SIAM International Conference on Data Mining*, 2001.

[23] J. Kim and J. H. Kim. Difference-based clustering of short time-course microarray data with replicates. *BMC Bioinformatics*, 8:253, 2007.

[24] J. Li. Mixture models. Lecture notes. *Pennsylvania State University*, 2009.

[25] H. Liu, S. Tarima, A. S. Borders, T. V. Getchell, M. L. Getchell, and A. J. Stromberg. Quadratic regression analysis for gene discovery and pattern recognition for non-cyclic short time-course mi-

croarray experiments. *BMC Bioinformatics*, 6:106–122, 2005.

[26] T. Liu, N. Lin, N. Shi, and B. Zhang. Information criterion-based clustering with order-restricted candidate profiles in short time-course microarray experiments. *BMC Bioinformatics*, 10:146–165, 2009.

[27] Y. Luan and H. Li. Clustering of time-course gene expression data using a mixed-effects model with B-splines. *Bioinformatics*, 19:474–482, 2003.

[28] P. Ma, C. I. Castillo-Davis, W. Zhong, and J. S. Liu. A data-driven clustering method for time course gene expression data. *Nucleic Acids Res*, 34:1261–1269, 2006.

[29] K. McCormick, R. Shrivastava, and L. Liao. Slopeminer: An improved method for mining subtle signals in time course microarray data. In F. P. Preparata, X. Wu, and J. Yin, editors, *Frontiers in Algorithmics*, volume 5059 of *Lecture Notes in Computer Science*, pages 28–34. Springer, 2008.

[30] C. S. Moller-Levet, K. H. Cho, and O. Wolkenhauer. Microarray data clustering based on temporal variation: FCV with TSD preclustering. *Appl Bioinformatics*, 2:35–45, 2003.

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

[32] S. D. Peddada, E. K. Lobenhofer, L. Li, C. A. Afshari, C. R. Weinberg, and D. M. Umbach. Gene selection and clustering for time-course and dose-response microarray experiments using order-restricted inference. *Bioinformatics*, 19:834–841, May 2003.

[33] S. Phan, F. Famili, Z. Tang, Y. Pan, Z. Liu, J. Ouyang, A. Lenferink, and M. M.-C. O'Connor. A novel pattern based clustering methodology for time-series microarray data. *Int J Comput Math*, 84(5):585–597, 2007.

[34] T. L. Phang, M. C. Neville, M. Rudolph, and L. Hunter. Trajectory clustering: a non-parametric method for grouping gene expression time courses, with applications to mammary development. *Pac Symp Biocomput*, 8:351–362, 2003.

[35] J. Qian, M. Dolled-Filhart, J. Lin, H. Yu, and M. Gerstein. Beyond synexpression relationships: Local clustering of time-shifted and inverted gene expression profiles identifies new, biologically relevant interactions. *J Mol Biol*, 314:1053–1066, 2001.

[36] M. F. Ramoni, P. Sebastiani, and I. S. Kohane. Cluster analysis of gene expression dynamics. *Proc Natl Acad Sci USA*, 99:9121–9126, 2002.

[37] J. M. Ross, C. Fan, M. D. Ross, T.-H. Chu, Y. Shi, L. Kaufman, W. Zhang, M. E. Klotman, and P. E. Klotman. HIV-1 infection initiatites an inflammatory cascade in human renal tubular epithelial cells. *J Acquir Immune Defic Syndr*, 42(1):1–11, 2006.

[38] L. Sacchi, R. Bellazzi, C. Larizza, P. Magni, T. Curk, U. Petrovic, and B. Zupan. TA-clustering: cluster analysis of gene expression profiles through Temporal Abstractions. *Int J Med Inform*, 74:505–517, 2005.

[39] D. Sahoo, D. L. Dill, R. Tibshirani, and S. K. Plevritis. Extracting binary signals from microarray time-course data. *Nucleic Acids Res*, 35:3705–3712, 2007.

[40] A. Schliep, A. Schonhuth, and C. Steinhoff. Using hidden Markov models to analyze gene expression time course data. *Bioinformatics*, 19 Suppl 1:i255–263, 2003.

[41] R. B. Stoughton. Applications of DNA microarrays in biology. *Annu Rev Biochem*, 74:53–82, 2005.

[42] X. Wang, M. Wu, Z. Li, and C. Chan. Short time-series microarray analysis: Methods and challenges. *BMC Syst Biol*, 2:58–63, 2008.

[43] S. Whiteson, P. Stone, K. O. Stanley, R. Miikkulainen, and N. Kohl. Automatic feature selection in neuroevolution. In *GECCO '05: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation*, pages 1225–1232, New York, NY, USA, 2005. ACM.

[44] M. L. Whitfield, G. Sherlock, A. J. Saldanha, J. I. Murray, C. A. Ball, K. E. Alexander, J. C. Matese, C. M. Perou, M. M. Hurt, P. O. Brown, and D. Bosteon. Identification of genes periodically expressed in the human cell cycle and their expression in tumors. *Mol Biol Cell*, 13(6):1977–2000, 2002.

[45] C. J. Wolfe, I. S. Kohane, and A. J. Butte. Systematic survey reveals general applicability of "guilt-by-association" within gene coexpression networks. *BMC Bioinformatics*, 6(227), 2005.

[46] F. X. Wu, W. J. Zhang, and A. J. Kusalik. Dynamic model-based clustering for time-course gene expression data. *J Bioinform Comput Biol*, 3:821–836, 2005.

[47] E. P. Xing, M. I. Jordan, and R. M. Karp. Feature selection for high-dimensional genomic microarray data. In *ICML '01: Proceedings of the Eighteenth International Conference on Machine Learning*, pages 601–608, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[48] K. Y. Yeung, C. Fraley, A. Murua, A. E. Raftery, and W. L. Ruzzo. Model-based clustering and data transformations for gene expression data. *Bioinformatics*, 17:977–987, 2001.

[49] S. G. Yi, Y. J. Joo, and T. Park. Rank-based clustering analysis for the time-course microarray data. *J Bioinform Comput Biol*, 7:75–91, 2009.

[50] S. Zhong and J. Ghosh. A unified framework for model-based clustering. *J Mach Learn Res*, 4:1001–1037, 2003.