Hindawi Publishing Corporation
EURASIP Journal on Bioinformatics and Systems Biology
Volume 2009, Article ID 195712, 12 pages
doi:10.1155/2009/195712

*Research Article*

# Clustering of Gene Expression Data Based on Shape Similarity

**Travis J. Hestilow**[1] **and Yufei Huang**[1, 2]

[1] *Department of Electrical and Computer Engineering, The University of Texas at San Antonio, San Antonio, TX 78249, USA*
[2] *Greehey Children's Cancer Research Institute, University of Texas Health Science Center at San Antonio, TX 78229, USA*

Correspondence should be addressed to Yufei Huang, yufei.huang@utsa.edu

Received 4 August 2008; Revised 8 January 2009; Accepted 27 January 2009

Recommended by Erchin Serpedin

A method for gene clustering from expression profiles using shape information is presented. The conventional clustering approaches such as K-means assume that genes with similar functions have similar expression levels and hence allocate genes with similar expression levels into the same cluster. However, genes with similar function often exhibit similarity in signal shape even though the expression magnitude can be far apart. Therefore, this investigation studies clustering according to signal shape similarity. This shape information is captured in the form of normalized and time-scaled forward first differences, which then are subject to a variational Bayes clustering plus a non-Bayesian (Silhouette) cluster statistic. The statistic shows an improved ability to identify the correct number of clusters and assign the components of cluster. Based on initial results for both generated test data and *Escherichia coli* microarray expression data and initial validation of the *Escherichia coli* results, it is shown that the method has promise in being able to better cluster time-series microarray data according to shape similarity.

## 1. Introduction

Investigating the genetic structure and metabolic functions of organisms is an important yet demanding task. Genetic actions, interactions, how they control and are controlled, are determined, and/or inferred by data from many sources. One of these sources is time-series microarray data, which measure the dynamic expression of genes across an entire organism. Many methods of analyzing this data have been presented and used. One popular method, especially for time-series data, is gene-based profile clustering [1]. This method groups genes with similar expression profiles in order to find genes with similar functions or to relate genes with dissimilar functions across different pathways occurring simultaneously.

There has been much work on clustering time-series data and clustering can be done based on either similarity of expression magnitude or the shape of expression dynamics. Clustering methods include hierarchical and partitional types (such as K-means, fuzzy K-means, and mixture modeling) [2]. Each method has its strengths and weaknesses. Hierarchical techniques do not produce clusters per se; rather, they produce trees or dendrograms. Clusters can be built from these structures by later cutting the output structure at various levels. Hierarchical techniques can be computationally expensive, require relatively smooth data, and/or be unable to "recover" from a poor guess; that is, the method is unable to reverse itself and recalculate from a prior clustering set. They also often require manual intervention in order to properly delineate the clusters. Finally, the clusters themselves must be well defined. Noisy data resulting in ill-defined boundaries between clusters usually results in a poor cluster set.

Partitional clustering techniques strive to group data vectors (in this case, gene expression profiles) into clusters such that the data in a particular cluster are more similar to each other than to data in other clusters. Partitional clustering can be done on the data itself or on spline representations of the data [3, 4]. In either case, square-error techniques such as K-means are often used. K-means is computationally efficient and can always find the global minimum variance. However, it must know the number of clusters in advance; there is no provision for determining an unknown number of clusters other than repeatedly testing the algorithm with different cluster numbers, which for large datasets can be very time consuming. Further, as is the case with hierarchical methods, K-means is best suited for clusters which are compact and well separated; it performs poorly with overlapping clusters. Finally, it is sensitive to noise and has no provision for accounting for such noise through a probabilistic model or the like. A related technique, fuzzy K-means, attempts to mimic the idea of posterior cluster membership probability through a concept of "degree of membership." However, this method is not computationally efficient and requires at least an a priori estimate of the degree of membership for each data point. Also, the number of clusters must be supplied a priori, or a separate algorithm must be used in order to determine the optimum number of clusters. Another similar method is agglomerative clustering [5]. Model-based techniques go beyond fuzzy K-means and actually attempt to model the underlying distributions of the data. The methods maximize the likelihood of the data given the proposed model [4, 6].

More recently, much study has been given toward clustering based on expression profile shape (or trajectory) rather than absolute levels. Kim et al. [7] show that genes with similar function often exhibit similarity in signal shape even though the expression magnitude can be far apart. Therefore, expression shape is a more important indication of similar gene functions than expression magnitude.

The same clustering methods mentioned above can be used based on shape similarity. An excellent example of a tree-based algorithm using shape-similarity as a criterion can be found in [8]. While the results of this investigation proved fruitful, it should be noted that the data used in the study resulted in well-defined clusters. Further, the clustering was done manually once the dendrogram was created. Möller-Levet et al. [9] used fuzzy K-means to cluster time-series microarray data using shape similarity as a criterion. However, the number of clusters was known beforehand; no separate optimization method was used in order to find the proper number of clusters. Balasubramaniyan et al. [10] used a similarity measure over time-shifted profiles to find local (short-time scale) similarities. Phang et al. [11] used a simple (+/0/−) shape decomposition and used a nonparametric Kruskal-Wallis test to group the trajectories. Finally, Tjaden [12] used a K-means related method with error information included intrinsically in the algorithm.

A common difficulty with these approaches is to determine the optimal number of clusters. There have been numerous studies and surveys over the years aimed at finding optimal methods for unsupervised clustering of data; for example, [13–20]. Different methods achieve different results, and no single method appears to be optimal in a global sense. The problem is essentially a model selection problem. It is well known that the Bayesian methods provide the optimal framework for selecting models, though a complete treatment is analytically intractable for most cases. In this paper, a Bayesian approach based on the Variational Bayes Expectation Maximization (VBEM) algorithm is proposed to determine the number of clusters and better performance than MDL and BIC criterion has been demonstrated.

In this study, the goal was to find clusters of genes with similar functions; that is, coregulated genes using time-series microarray data. As a result, we choose to cluster genes based on signal shape information. Particularly, signal shape information is derived from the normalized time-scaled forward first differences of the time-sequence data. This information is then forwarded to a Variational Bayes Expectation Maximization algorithm (VBEM, [21]), which performs the clustering. Unlike K-means, VBEM is a probabilistic method, which was derived based on the Bayesian statistical framework and has shown to provide better performance. Further, when paired with an external clustering statistic such as the Silhouette statistic [22], the VBEM algorithm can also determine the optimal number of clusters.

The rest of the paper is organized as follows. In Section 2 the problem is discussed in more detail, the underlying model is developed, and the algorithm is presented. In Section 3 the results of our evaluation of the algorithm against both simulated and real time-series data are shown. Also presented are comparisons between the algorithm and K-means clustering, both methods using several different criteria for making clustering decisions. Conclusions are summarized in Section 4. Finally, Appendices A, B, and C present a more detailed derivation of the algorithm.

## 2. Method

*2.1. Problem Statement and Method.* Given the microarray datasets of $G$ genes, $\mathbf{x}_g \in \mathbb{R}^{N\times 1}$ for $(g = 1, 2, 3, \dots, G)$, where $N$ is the number of time points, that is, the columns in the microarray, it is desired to cluster the gene expressions based on signal shape. The clustering is not known a priori; therefore not only must individual genes be assigned to relevant clusters, but the number of clusters themselves must also be determined.

The clustering is based on expression-level shape rather than magnitude. The shape information is captured by the first-order time difference. However, since the gene expression profiles were obscured by the varying levels manifested in the data, the time difference must be obtained on the expression levels with the same scale and dynamic range. Motivated by the observations, the proposed algorithm has three steps. In the first step, the expression data is rescaled. In the second step, the signal shape information is captured by calculating the first-order time difference. In the last step, clustering is performed on the time-difference data using a Variational Bayes Expectation Maximization (VBEM) algorithm. In the following, each step is discussed in detail.

*2.2. Initial Data Transformation.* Each gene sequence was rescaled by subtracting the mean value of each sequence from each individual gene, resulting in sequences with zero mean. This operation was intended to mitigate the widely different magnitudes and slopes in the profile data. By resetting all genes to a zero-mean sequence, the overall shape of each sequence could be better identified without the complication of comparing genes with different magnitudes.

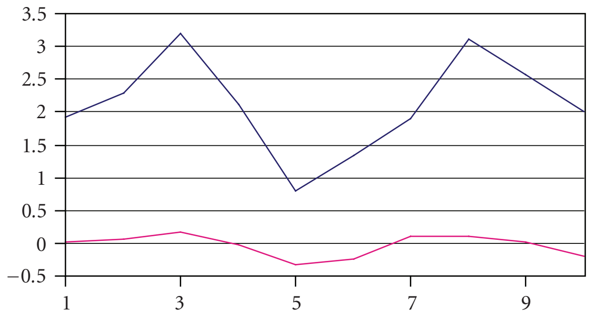

FIGURE 1: Dissimilar expression levels with similar shape.

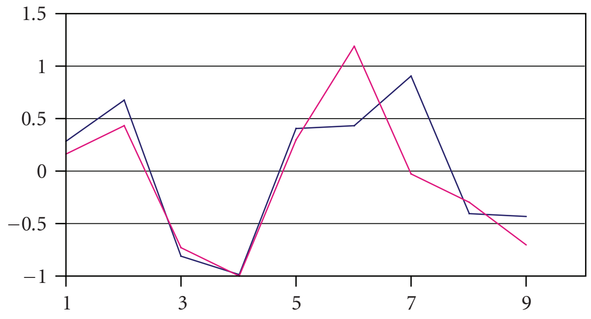

FIGURE 2: Normalized differences: the same two sequences after transformations.

After this, the resulting sequences were then normalized such that the maximum absolute value of the sequence was 1. Gene expression between related genes can result in a large change or a small; if two genes are related, that relationship should be recoverable regardless of the amplitude of change. By renormalizing the data in this manner, the amplitudes of both large-change and small-change genes were placed into the same order of magnitude.

Mathematically, the above operation can be expressed by

$$\mathbf{z}_g = \frac{\mathbf{x}_g - \mu_{x_g}}{\max\left(\operatorname{abs}(\mathbf{x}_g - \mu_{x_g})\right)}, \tag{1}$$

where $\mu_{x_g}$ represents the mean of $\mathbf{x}_g$.

*2.3. Extraction of Shape Information and Time Scaling.* To extract shape information of time-varying gene expression, the derivative of the expression trajectory is considered. Since we are dealing with discrete sequences, differences must be used rather than analytical derivatives. To characterize the shape of each sequence, a simple first-difference scheme was used, this being the magnitude difference of the succeeding point and the point under consideration, divided by the time difference between those points. The data was taken nonuniformly over a period of approximately 100 minutes, with sample times varying from 7 to 50 minutes. As the transformation in (1) already scales the data to a range of $[-1, 1]$, further compressing that scale by nearly 2 orders of magnitude over some time stretches was deemed neither prudent nor necessary. Therefore, the time difference was scaled in hours to prevent this unneeded range compression. The resulting sequences were used as data for clustering.

Mathematically, this operation can be written as

$$y_{g,k} = \frac{z_{g,k+1} - z_{g,k}}{t_{g,k+1} - t_{g,k}}, \quad k = 1 \cdots N-1, \tag{2}$$

where $t_g$ is the length-$N$ vector of time points associated with gene $g$, $\mathbf{z}_g$ is the vector of transformed time-series data (from (1)) associated with gene $g$, and $\mathbf{y}_g$ is the resulting vector of first differences associated with gene $g$.

Figure 1 shows an example pair of sequences using contrived data. These two sequences are visually related in shape, but their mean values are greatly different. A K-means clustering would place these two sequences in different clusters. By transforming the data, the similarity of the two sequences is enhanced, and the clustering algorithm can then place them in the same cluster. Figure 2 shows the original two sequences after data transformation.

*2.4. Clustering.* Once the sequence of first differences was calculated for each gene, clustering was performed on $\mathbf{y}$, the first-order difference. To this end, a VBEM algorithm was developed. Before presenting that development, a general discussion of VBEM is in order.

An important problem in Bayesian inference is determining the best model for a set of data from many competing models. The problem itself can be stated fairly compactly. Given a set of data $\mathbf{y}$, the marginal likelihood of that data given a particular model $m$ can be expressed as

$$p(\mathbf{y} \mid m) = \int p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta} \mid m) d\mathbf{x}\, d\boldsymbol{\theta}, \tag{3}$$

where $\mathbf{x}$ and $\boldsymbol{\theta}$ are, respectively, the latent variables and the model parameters. The integration is taken over both variables and parameters in order to prevent overfitting, as a model with many parameters would naturally be able to fit a wider variety of datasets than a model with few parameters.

Unfortunately, this integral is not easily solved. The VBEM method approximates this by introducing a free distribution, $q(\mathbf{x}, \boldsymbol{\theta})$, and taking the logarithm of the above integral. If $q(\mathbf{x}, \boldsymbol{\theta})$ has support everywhere that $p(\mathbf{x}, \boldsymbol{\theta} \mid \mathbf{y}, m)$ does, we can construct a lower bound to the integral using Jensen's inequality:

$$\begin{aligned} \ln p(\mathbf{y} \mid m) &= \ln \int p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta} \mid m) d\mathbf{x}\, d\boldsymbol{\theta} \\ &= \ln \int q(\mathbf{x}, \boldsymbol{\theta}) \frac{p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta} \mid m)}{q(\mathbf{x}, \boldsymbol{\theta})} d\mathbf{x}\, d\boldsymbol{\theta} \\ &\geq \int q(\mathbf{x}, \boldsymbol{\theta}) \ln \frac{p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta} \mid m)}{q(\mathbf{x}, \boldsymbol{\theta})} d\mathbf{x}\, d\boldsymbol{\theta}. \end{aligned} \tag{4}$$

Maximizing this lower bound with respect to the free distribution $q(\mathbf{x}, \boldsymbol{\theta})$ results in $q(\mathbf{x}, \boldsymbol{\theta}) = p(\mathbf{x}, \boldsymbol{\theta} \mid \mathbf{y}, m)$, the joint posterior. Since the normalizing constant is not known, this posterior cannot be calculated exactly. Therefore another simplification is made. The free distribution $q(\mathbf{x}, \boldsymbol{\theta})$

is assumed to be factorable, that is, $q(\mathbf{x},\boldsymbol{\theta}) = q(\mathbf{x})q(\boldsymbol{\theta})$. The inequality then becomes

$$\begin{aligned}\ln p(\mathbf{y} \mid m) &\geq \int q(\mathbf{x})q(\boldsymbol{\theta}) \ln \frac{p(\mathbf{y},\mathbf{x},\boldsymbol{\theta} \mid m)}{q(\mathbf{x})q(\boldsymbol{\theta})} d\mathbf{x}\, d\boldsymbol{\theta} \\ &= \mathcal{F}(q(\mathbf{x}), q(\boldsymbol{\theta})).\end{aligned} \tag{5}$$

Maximizing this functional $\mathcal{F}$ is equivalent to minimizing the KL distance between $q(\mathbf{x})q(\boldsymbol{\theta})$ and $p(\mathbf{x},\boldsymbol{\theta} \mid \mathbf{y}, m)$. The distributions $q(\mathbf{x})$ and $q(\boldsymbol{\theta})$ are coupled and must be iterated until they converge.

With the above discussion in mind, we now develop the model that our VBEM algorithm is based on. Given $K$ clusters in total, we can let $C_g \in \{1, 2, \dots, k\}$ denote the cluster number of gene $g$. Then, we assume that, given $C_g = k$, the expression level for gene $g$ follows a Gaussian distribution, that is,

$$p(y_g \mid C_g = k, \mathbf{m}_{1:k}, \mathbf{s}^2_{1:k}) = \mathcal{N}(\mathbf{m}_k, \operatorname{diag}(\mathbf{s}_k{}^2)), \tag{6}$$

where $\mathbf{m}_k = [m_{k_1}, m_{k_2}, \dots, m_{k_N}]^T$ is the mean and $\mathbf{s}_k{}^2 = [\mathbf{s}_{k_1}{}^2, s_{k_2}{}^2, \dots, s_{k_N}{}^2]^T$ is the variance of the $k$th Gaussian cluster. Since both $\mathbf{m}_k$ and $\mathbf{s}_k{}^2$ are unknown parameters, a Normal-Inverse-Gamma prior distribution is assigned as

$$p(\mathbf{m}_k, \mathbf{s}_k{}^2) = \prod_{j=1}^{N} \mathcal{N}\left(0, \frac{s_{i,j}{}^2}{k}\right) \mathrm{IG}\left(s_{i,j}{}^2 \mid \frac{a_0}{2}, \frac{b_0}{2}\right), \tag{7}$$

where $k$, $a_0$, and $b_0$ are the known parameters of the prior distribution. Furthermore, a multinomial prior is assigned for the cluster number $C_g$ as

$$p(C_g = k \mid \mathbf{L}) = L_k, \tag{8}$$

where $L_k$ is the prior probability that gene $g$ belongs to $k$th cluster $k$ and $\sum_{k=1}^{K} L_k = 1$. $L_k$ further assumes a priori the Dirichlet distribution

$$p(L_1, L_2, \dots, L_k) = \operatorname{Dir}(a_1, \dots, a_k), \tag{9}$$

where $a_1 \cdots a_k$ are the known parameters of the distribution. Given the transformed expressions of $G$ genes, $\mathbf{y} = [y_1, y_2, \dots, y_G]^T$, the stated two tasks are equivalent to estimating $K$, the total number of clusters, and $C_g$ for all $G$ genes.

A Bayesian framework is adopted for estimating both $K$ and $C_g$, which are calculated by the maximum a posteriori criterion as

$$\begin{aligned} K_{\max} &= \arg\max_{K} p(\mathbf{y} \mid H = K), \\ C_{g,\max} &= \arg\max_{k} p(C_g = k \mid \mathbf{y}), \quad k \in 1, \dots, K_{\max}, \end{aligned} \tag{10}$$

where $p(\mathbf{y} \mid H = k)$ is the marginal likelihood given the model $H$ has $K$ clusters, and $p(C_g = k \mid \mathbf{y})$ is the *a posteriori* probability of $C_g$ when the total number of clusters is $K$.

Unfortunately, there are now multiple unknown nuisance parameters at this point: $m_k$, $s_k{}^2$, $a$, $b$, $k$, and $L$ all still need to be found. To do so requires a marginalization procedure over all the unknowns, which is intractable for unknown cluster id $C_g$. Therefore, a VBEM scheme is adopted for estimating the necessary distributions.

*2.5. VBEM Algorithm.* Given the development above, $p(\mathbf{y} \mid H = K)$ can be expressed as

$$p(\mathbf{y} \mid H = k) = \sum_{C_g} \int p(\mathbf{y} \mid C_g = k, \boldsymbol{\theta}) p(C_g) p(\boldsymbol{\theta}) d\boldsymbol{\theta}, \tag{11}$$

where $\boldsymbol{\theta}$ is the vector of unknown parameters $m_k$, $s_k{}^2$, $a$, $b$, $k$, and $L$. Notice the summation in (11) is NP hard, whose complexity increases exponentially with the number of genes. We therefore resort to approximate this integration by variational EM. First, a lower bound is constructed for the expression in (11). The ultimate aim is to maximize this lower bound. The expression for the lower bound can be written

$$\begin{aligned} &\ln p(\mathbf{y} \mid H = k) \\ &\quad = \ln \sum_{C_g} \int p(\mathbf{y} \mid C_g, \boldsymbol{\theta}) p(C_g) p(\boldsymbol{\theta}) d\boldsymbol{\theta} \\ &\quad \geq \ln \int \left[ \sum_{C_g} q(C_g) \ln \frac{p(\mathbf{y}, C_g \mid \boldsymbol{\theta})}{q(C_g)} + \ln \frac{p(\boldsymbol{\theta})}{q(\boldsymbol{\theta})} \right] d\boldsymbol{\theta}, \end{aligned} \tag{12}$$

where as above the inequality derives by use of Jensen's inequality. The free distributions $q(C_g)$ and $q(\boldsymbol{\theta})$ are introduced as approximations to the unknown distributions $p(C_g \mid \mathbf{y})$ and $p(\boldsymbol{\theta} \mid \mathbf{y})$. The $q(\cdot)$ distributions are chosen so as to maximize the lower bound. Using variational derivatives and an iterative coordinate ascent procedure, we find

*VBE Step:*

$$q^{j+1}(C_g) = \frac{1}{Z_{C_g}} \exp\left[\int q^{(j)}(\boldsymbol{\theta}) \ln p(C_g, \mathbf{y} \mid \boldsymbol{\theta})\right] d\boldsymbol{\theta}; \tag{13}$$

*VBM Step:*

$$q^{j+1}(\boldsymbol{\theta}) = \frac{1}{Z_{\boldsymbol{\theta}}} \exp\left[\sum_{C_g} q^{(j+1)}(C_g) \ln p(C_g, \mathbf{y} \mid \boldsymbol{\theta})\right], \tag{14}$$

where $j$ and $j+1$ are iterations and $Z(\cdot)$ are normalizing constants to be determined. Because of the integration in (13), $q(\boldsymbol{\theta})$ must be chosen carefully in order to have an analytic expression. By choosing $q(\boldsymbol{\theta})$ as a member of the exponential family, this condition is satisfied. Note $q(\boldsymbol{\theta})$ is an approximation to the posterior distribution $q(\boldsymbol{\theta} \mid \mathbf{y})$ and therefore can be used to obtain the estimate of $\boldsymbol{\theta}$.

*2.6. Summary of VBEM Algorithm.* The VBEM algorithm is summarized as follows:

```
(1) Initialization
    (i) Initialize $m_k$, ${s_k}^2$, $\mathbf{a}$, $\mathbf{b}$, $\mathbf{k}$, and $\mathbf{L}$.
Iterate until lower bound converges enumerate
(2) VBE Step:
    (i) for $k = 1 : K,\ g = 1 : G$,
    (ii) calculate $q(C_g = k)$ using (A.1) in Appendix A,
    (iii) end $g,\ k$.
(3) VBM Step:
    (i) for $k = 1 : K$,
    (ii) calculate $q(\boldsymbol{\theta})$ using (B.1) in Appendix B,
    (iii) End $k$.
(4) Lower bound:
    (i) calculate $\mathbf{F}(q(C_g),\ q(\boldsymbol{\theta}))$ using (C.1) in Appendix C.
    End iteration.
```

*2.7. Choice of the Optimum Number of Clusters.* The Bayesian formulation of (11) suggests using the number of clusters that maximize the marginal likelihood, or in the context of VBEM, the lower bound $F(\cdot)$. Instead of solely basing the determination of the number of clusters using $F(\cdot)$, 4 different criteria are investigated in this work: (a) lower bound $F(\cdot)$ used within the VBEM algorithm (labelled KL), (b) the Bayes Information Criterion [23], (c) the Silhouette statistic performed on clusters built from transformed data, and (d) the Silhouette statistic performed on clusters built from raw data. The VBEM lower bound $F(\cdot)$ is discussed above; the BIC and Silhouette criteria are discussed below.

*2.8. Bayes Information Criterion (BIC).* The Bayes Information Criterion (BIC, [23]) is an asymptotic approximation to the *Bayes Factor*, which itself is an average likelihood ratio similar to the maximum likelihood ratio. As the Bayes Factor is often a difficult calculation, the BIC offers a less-intensive approximation. Subject to the assumptions of large data size and exponential-family prior distributions, maximizing the BIC is equivalent to maximizing the integrated likelihood function. The BIC can be written as

$$\mathrm{BIC} = 2\ln p(\mathbf{x} \mid \boldsymbol{\theta}) - k\ln(n), \tag{15}$$

where $p(\mathbf{x} \mid \boldsymbol{\theta})$ is the likelihood function of data $x$ given parameters $\boldsymbol{\theta}$, $k$ is the size (dimensionality) of parameter set $\boldsymbol{\theta}$, and $n$ is the sample size. The term $-k\ln(n)$ is a penalty term discouraging more complex models.

*2.9. Silhouette Statistic.* The Silhouette statistic (Sil, [22]) uses the squared difference between a data vector and all other data vectors in all clusters. For any particular data vector $\mathbf{v}$ belonging to cluster $A$, let $a_{\mathbf{v}}$ be the average squared difference between data vector $\mathbf{v}$ and all other vectors in cluster $A$. Let $b_{\mathbf{v}}$ be the minimum average squared distance between data vector $\mathbf{v}$ and all other vectors of cluster $B$, $B \neq A$. Then the Silhouette statistic for data vector $\mathbf{v}$ is

$$\mathrm{Sil}(\mathbf{v}) = \frac{b_{\mathbf{v}} - a_{\mathbf{v}}}{\max\{a_{\mathbf{v}}, b_{\mathbf{v}}\}}. \tag{16}$$

It is quickly seen that the range of this statistic is $[-1, 1]$. A value close to 1 means the data vector is very probably assigned to the correct cluster, while a value close to $-1$ means the data vector is very probably assigned to the wrong cluster. A value near 0 is a neutral evaluation.

## 3. Results

We illustrate the method using simulated expression data and with microarray data available online.

*3.1. Simulation Study.* In order to test the ability of VBEM to properly cluster data of similar shape but dissimilar mean level, and scale, several datasets were constructed. These datasets were intended to appear as would a set of time-series microarray data. Each consisted of 5 data points in a vector, corresponding to what might be seen from a microarray from a single gene over 5-time samples. Identical assumptions were used to produce these datasets; namely, that the inherent clusters within the data were based upon a mean vector of values for a particular cluster, that each cluster may have subclusters exhibiting a mean shift and/or a scale change from the mean vector, and that the data within a cluster randomly varied about that mean vector (plus any mean shift and scale change). All sets of sample data shared the characteristics shown in Table 1. For example, a test "gene" of cluster "dms" would be a random length-5 vector, drawn from a Gaussian distribution with a mean of $[2.0\ \ -2.0\ \ 0.0\ \ 0.0\ \ 0.0]$ and a particular standard deviation (defined below). This random vector would then be scaled by 0.25 and shifted in value by $-1.25$.

The datasets constructed from these basis vectors differed in number of data vectors per subcluster (and thus the total number of data vectors), and the standard deviation used to vary the individual vector values about their corresponding basis vectors. Generally speaking, the standard deviation vectors were constructed to be approximately 25% of the mean vector for the "low-noise" sets, and approximately 50% of the mean vector for the "high-noise" sets.

*3.2. "Low-Noise" Test Datasets.* Two datasets were constructed using standard deviation vectors approximately 25% of the relevant mean vector. Table 2 shows the standard deviation vectors used. Each subcluster in Table 1 was replicated several times, randomly varying about the mean vector in a Gaussian distribution with a standard deviation as shown in Table 2. Test set 1 had 5 replicates per subcluster (e.g., a1–a5, cs1–cs5), resulting in a total set $N = 55$ data vectors. Test set 2 had 99 replicates per subcluster, resulting in a total set $N = 1089$ data vectors.

Table 1: Basis vectors for clusters in sample datasets.

| Cluster | Subcluster | Mean vector | Mean shift | Scale factor |
|---|---|---|---|---|
| a | a | [0.5 0.5 0.5 0.5 2.0] | 0 | 1 |
| b | b | [0.5 2.0 2.0 −2.0 −2.0] | 0 | 1 |
| | bm | | −1.25 | 1 |
| c | c | [0.0 2.0 0.0 2.0 0.0] | 0 | 1 |
| | cs | | 0 | 0.25 |
| d | d | [2.0 −2.0 0.0 0.0 0.0] | 0 | 1 |
| | dms | | −1.25 | 0.25 |
| e | e | [−2.0 0.0 0.0 0.0 −2.0] | 0 | 1 |
| | em | | −1.25 | 1 |
| | es | | 0 | 0.25 |
| | ems | | −1.25 | 0.25 |

Table 2: Standard deviation vectors for clusters in "low-noise" sample datasets.

| Cluster | Standard deviation vector |
|---|---|
| a | [0.1 0.1 0.1 0.1 0.5] |
| b | [0.1 0.5 0.5 0.5 0.5] |
| c | [0.1 0.5 0.1 0.5 0.1] |
| d | [0.5 0.5 0.1 0.1 0.1] |
| e | [0.5 0.1 0.1 0.1 0.5] |

Table 3: Standard deviation vectors for clusters in "high-noise" sample datasets.

| Cluster | Standard deviation vector |
|---|---|
| a | [0.2 0.2 0.2 0.2 1.0] |
| b | [0.2 1.0 1.0 1.0 1.0] |
| c | [0.2 1.0 0.2 1.0 0.2] |
| d | [1.0 1.0 0.2 0.2 0.2] |
| e | [1.0 0.2 0.2 0.2 1.0] |

### 3.3. *"High-Noise" Test Datasets.*

Because of the need to test the robustness of the clustering and prediction algorithms in the presence of higher amounts of noise, six datasets were constructed using standard deviation vectors approximately 50% of the relevant mean vector. Table 3 shows the standard deviation vectors used. As with the "low-noise" sets, each subcluster in Table 1 was replicated several times, randomly varying about the mean vector in a Gaussian distribution, this time with a standard deviation as shown in Table 3. Table 4 shows the number of replicates produced for each dataset. For the test data, an added transformation step was accomplished that would normally not be performed on actual data. Since the test data was produced in already clustered form, the vectors (rows) were randomly shuffled to break up this clustering.

### 3.4. *Test Types and Evaluation Measures.*

To evaluate the ability of VBEM to properly cluster the datasets, two test sequences were conducted. First, the data was clustered using VBEM in a "controlled" fashion; that is, the number of clusters was assumed to be known and passed to the algorithm. Second, the algorithm was tested in an "uncontrolled" fashion; that is, the number of clusters was unknown, and the algorithm had to predict the number of clusters given the data. During the uncontrolled tests, a K-means algorithm was also run against the data as a comparison.

Table 4: Subcluster replicates and total vector sizes for "high-noise" datasets.

| Test set | Total replicates | Total $N$ |
|---|---|---|
| 3 | 5 | 55 |
| 4 | 9 | 99 |
| 5 | 30 | 330 |
| 6 | 50 | 550 |
| 7 | 70 | 770 |
| 8 | 99 | 1089 |

The VBEM algorithm as currently implemented requires an initial (random) probability matrix for the distribution of genes to clusters, given a value for $K$. Therefore, for each dataset, 55 trials were conducted, each trial having a different initial matrix.

Also, each trial begins with an initial clustering of genes. As currently implemented, this initialization is performed using a K-means algorithm. The algorithm attempts to cluster the data such that the sum of squared differences between data within a cluster is minimized. Depending on the initial starting position, this clustering may change. In MATLAB, the built-in K-means algorithm has several options available to include how many different trials (from different starting points) are conducted to produce a "minimum" sum-squared distance, how many iterations are allowed per trial to reach a stable clustering, and how clusters that become "empty" during the clustering process are handled. For these tests, the K-means algorithm conducted 100 trials of its own per initial probability matrix (and output the clustering with the smallest sum-squared distance), had a limit of 100 iterations, and created a "singleton" cluster when a cluster became empty.

As mentioned above, the choice of optimum $K$ was conducted using four different calculations. The first used

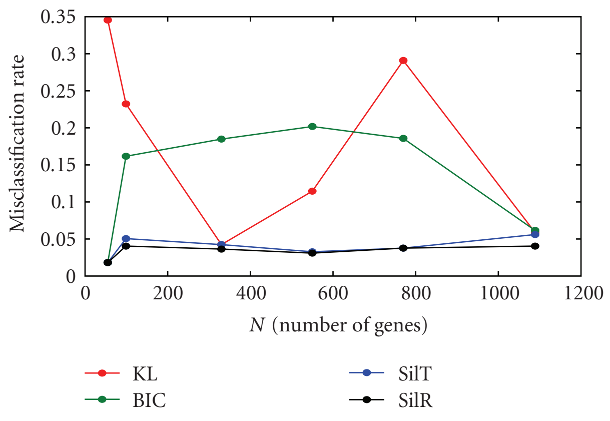

Figure 3: Misclassification rate versus $N$, high-noise data, $K$ fixed.

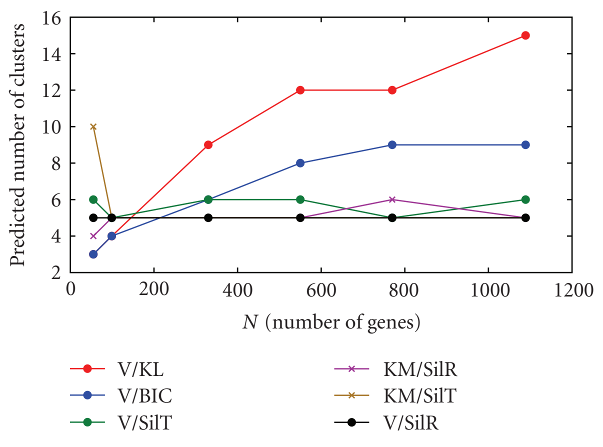

Figure 4: $K$(pred) versus $N$, high-noise data.

the estimate for the VBEM lower bound, the second used the BIC equation. In both cases, the optimum $K$ for a particular trial was that which showed a decrease in value when $K$ was increased. This does not mean the values used to determine the optimum $K$ were the absolute maxima for the parameter within that trial; in fact, they usually were not. The overall optimum $K$ for a particular choice of parameter was the maximum value over the number of trials. The third and fourth criteria made use of the Silhouette statistic, one using the clusters of transformed data and one using the corresponding clusters of raw data. We used the built-in Silhouette function contained within MATLAB for our calculations. To find the optimum $K$, the mean Silhouette value for all data vectors in a clustering was calculated for each value of $K$. The value of $K$ for which the mean value was maximized was chosen as the optimum $K$.

To evaluate the actual clustering, a misclassification rate was calculated for each trial cluster. Since the "ground-truth" clustering was known a priori, this rate can be calculated as a sum of probabilities derived from the original data and the clustering results:

$$R_{\text{mi}} = \sum_{j=1}^{K} \sum_{k=1}^{K} p(C_j \mid C_k)\, p(C_k), \tag{17}$$

where $p(C_j \mid C_k)$ is the probability that computed cluster $C_j$ belongs to a priori cluster $C_k$ given that $C_k$ is in fact the correct cluster, and $p(C_k)$ is the probability of a priori cluster $C_k$ occurring. $R_{\text{mi}}$ refers to the misclassification rate using statistic $m$ (KL, BIC, both Silhouette) for trial $i$. This rate is in the range $[0, 1]$ and is equal to 1 only when the number of clusters is properly predicted and those calculated clusters match the a priori clusters. Thus, both under- and overprediction of clusters were penalized.

For the "controlled" test sequences, the combinations of VBEM + KL (V/KL), VBEM + BIC (V/BIC), VBEM + Silhouette (transformed data) (V/SilT), and VBEM + Silhouette (raw data) (V/SilR) all properly chose the optimum clustering for the two "low-noise" datasets, in all cases with no misclassification. For the six "high-noise" sets, V/KL and V/BIC were completely unable to choose the optimum clustering (lowest misclassification rate). In the case of V/SilT, the algorithm-chosen optimum was rarely the true optimum (2 out of 6 datasets). However, the chosen optimum was always very nearly optimal. Finally, V/SilR chose the optimum clustering 5 out of 6 datasets. The algorithm-chosen optimal clustering for both V/SilT and V/SilR showed a misclassification rate of 6 percent or less, while the misclassification rates for V/KL and V/BIC were often in the range of 15–35 percent. Figure 3 summarizes this data.

For the "uncontrolled" tests, the above 4 algorithms were tested with the number of clusters unknown. Further, K-means clustering with Silhouette statistic (KM/SilT and KM/SilR) was also conducted for comparison. The results for the 6 "high-noise" datasets are summarized below.

Figure 4 shows a summary plot of the predicted number of clusters $K$ versus dataset size $N$ for all combinations. Note that V/SilR correctly identified $K = 5$ for all datasets. Also note that KM/SilT, KM/SilR, and V/SilT predicted $K = 5$ or $K = 6$ for all datasets except for test set 3 ($N = 55$). However, even though V/SilR correctly identified $K = 5$ for this dataset, it had equivalent optimum values for $K = 7, 8, 10$, and 15. Given the poor performance of all combinations for this dataset, this suggests that for high-noise data such as this, $N = 55$ is insufficient to give good results.

V/KL and V/BIC both performed poorly with all datasets, in most cases overpredicting the number of clusters. As can be seen in Figure 4, this overprediction tended to increase with dataset size $N$. V/BIC resulted in a lower over-prediction than V/KL.

Figure 5 shows a summary plot of misclassification rate versus dataset size $N$ for the VBEM versus K-means comparison using Silhouette statistics only (both raw and difference). This plot shows the greater performance of V/SilR even more dramatically. While the misclassification rates for the KM/SilT, KM/SilR, and V/SilT were generally on the order of 10–20%, V/SilR was very stable, generally between 3-4%.

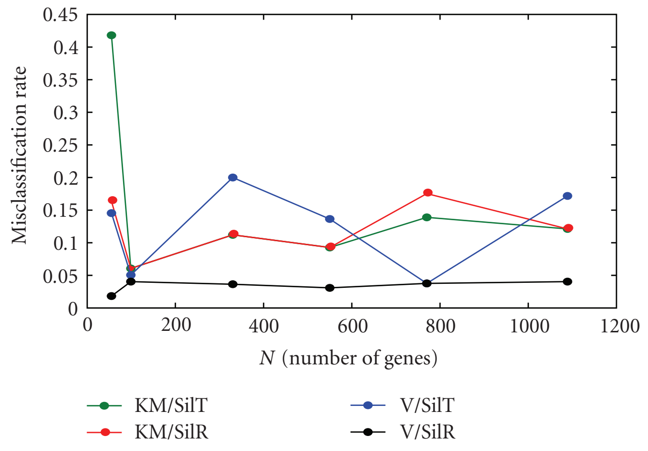

FIGURE 5: Misclassification rate versus $N$, high-noise data, $K$ unknown.

*3.5. Test Results Conclusion.* The VBEM algorithm can correctly cluster shape-based data even in the presence of fairly high amounts of noise, when paired with the Silhouette statistic performed on the raw data clusters (V/SilR). Further, V/SilR is robust in correctly predicting the number of clusters in noise. The misclassification rate is superior to K-means using Silhouette statistics, as well as VBEM using all other statistics. Because of this, it was expected that V/SilR would be the algorithm of choice for the experimental microarray data. However, to maintain comparison, all four VBEM/statistic algorithms were tested.

*3.6. Experimental E. Coli Expression Data.* The proposed approach for gene clustering on shape similarity was tested using time-series data from the University of Oklahoma E. coli Gene Expression Database resident at their Bioinformatics Core Facility (OUBCF) [24]. The exploration concentrated on the wild-type MG1655 strain during exponential growth on glucose. The data available consisted of 5 time-series log-ratio samples of 4389 genes.

The initial tests were run against genes identified as being from metabolic categories. Specifically, genes identified in the E. coli K-12 Entrez Genome database at the National Center for Biotechnology Information, US National Library of Medicine, National Institutes of Health (http://www.ncbi.nlm.nih.gov/) [25] (NIH) as being in categories C, G, E, F, H, I, and/or Q were chosen.

Because of the short-sequence lengths, any gene with even a single invalid data point was removed from the set. With only 5-time samples to work with in each gene sequence, even a single missing point would have significant ramifications in the final output. The final set of genes used for testing numbered 1309.

In implementing the VBEM algorithm, initial values for the algorithm were $a_0 = b_0 = 0.0002$. The algorithm was set to iterate until the change in lower bound decreased below $5 \times 10^{-2}$ or became negative (which required the prior iteration to be taken as the end value) or 200 iterations, whichever came first. The optimal number of clusters was arrived at by multiple runs of the algorithm at values of $K$, the predefined number of clusters, varying from 3 to 15. $K$ was chosen in the same manner as in the test data sequences.

Figure 6 shows a summary of the final result of the algorithm. Each subfigure shows the mean shapes clustered by the particular algorithm/statistic. As can be seen from the figure, V/KL resulted in an overclassification of structure in the data. The other three algorithms gave more consistent results. As a result of this, the V/KL clusters were removed from further analysis.

*3.7. Validation of E. Coli Expression Data Results.* We validated the results of our tests using Gene Ontology (GO) enrichment analysis. To this end, the genes used in the analysis were tagged with their respective GO categories and analyzed within each cluster for overrepresentation of certain categories versus the "background" level of the population (in this case, the entire set of metabolic genes used). Again, the Entrez Genome database at NIH was used for the GO annotation information. As most of the entries enriched were from the Biological Process portion of the ontology, the analysis was restricted to those terms.

To perform the analysis, the software package Cytoscape (http://www.cytoscape.org/) [26] was used. Cytoscape offers access to a wide variety of plug-in analysis packages, including a GO enrichment analysis tool, BiNGO, which stands for Biological Network Gene Ontology (http://www.psb.ugent.be/cbd/papers/BiNGO/) [27].

To evaluate the clusters, we modified an approach used by Yuan and Li [28] to score the clusters based on the information content and the likelihood of enrichment ($P$-value $< .05$). Unlike [28], however, a distance metric was not included in the calculations. Because of the large cluster sizes involved, such distance calculations would have exacted a high calculation overhead. Rather, the simpler approach of forming subclusters of adjacent enriched terms was chosen; that is, if two GO terms had a relationship to each other and were both enriched, they were placed in the same subcluster and their scores multiplied by the number of terms in the subcluster. Also, a large portion of the score of any term shared across more than one cluster was subtracted. This method rewarded large subclusters, while penalizing numerous small subclusters and overlapping terms.

The scoring equation for a cluster $C$, consisting of $k$ subclusters each of size $n_k$ is given as

$$\begin{aligned}\text{Score}_C = &\sum_{j=1}^{k}(n_j - 1)\sum_{i=1}^{n_j}\log\left(\Pr(t_{ij})\right)\log\left(p_{ij}\right)\\ &-\left(\frac{n-1}{n}\right)\sum_{\substack{t_k\in C_i\cap C_j\cap\\ \cdots\cap C_n}}\log\left(\Pr(t_k)\right)\log\left(p_k\right),\end{aligned} \tag{18}$$

where $\Pr(t_{ij})$ is the probability of GO term $t_{ij}$ being selected, $\log(\Pr(t_{ij}))$ is the negative of the information content of the GO term, and $p_{ij}$ is the $P$-value ($P < .05$) of the GO term $t_{ij}$. Large subclusters are rewarded by larger values of $n_k$. Subtracting 1 from $n_k$ compensates for the "baseline" score value; that is, the score a cluster would achieve if no terms were connected. The final term in the equation is the devaluation of any GO term shared by $n$ clusters.

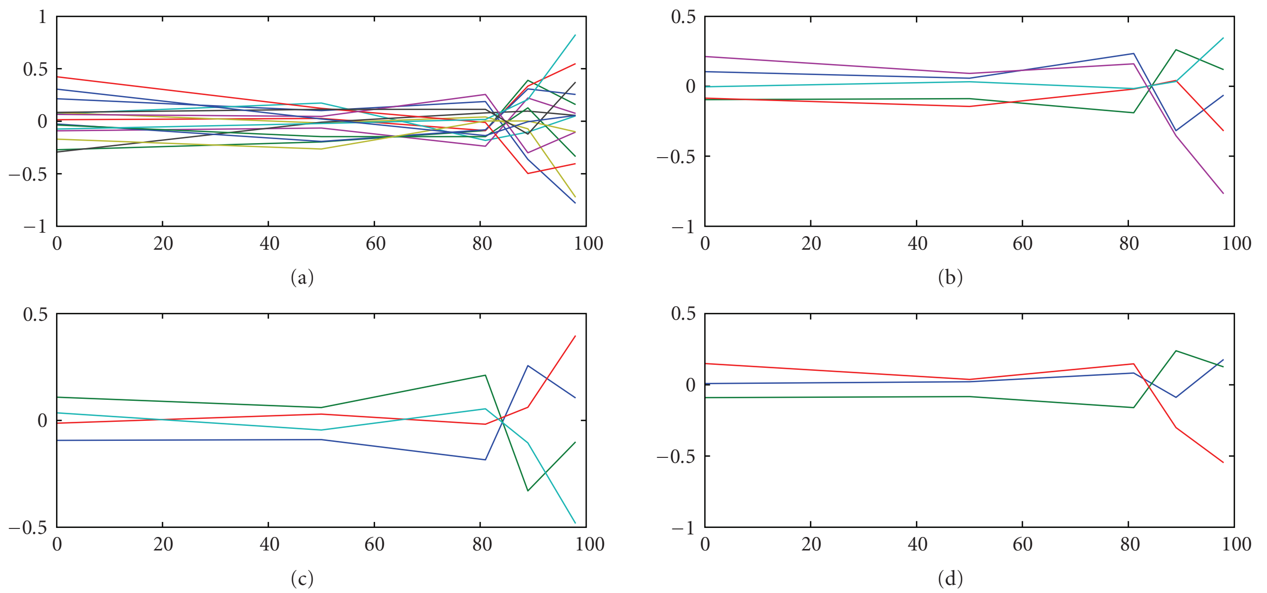

Figure 6: Mean data shapes. (a) V/KL, (b) V/BIC, (c) V/SilT, (d) V/SilR.

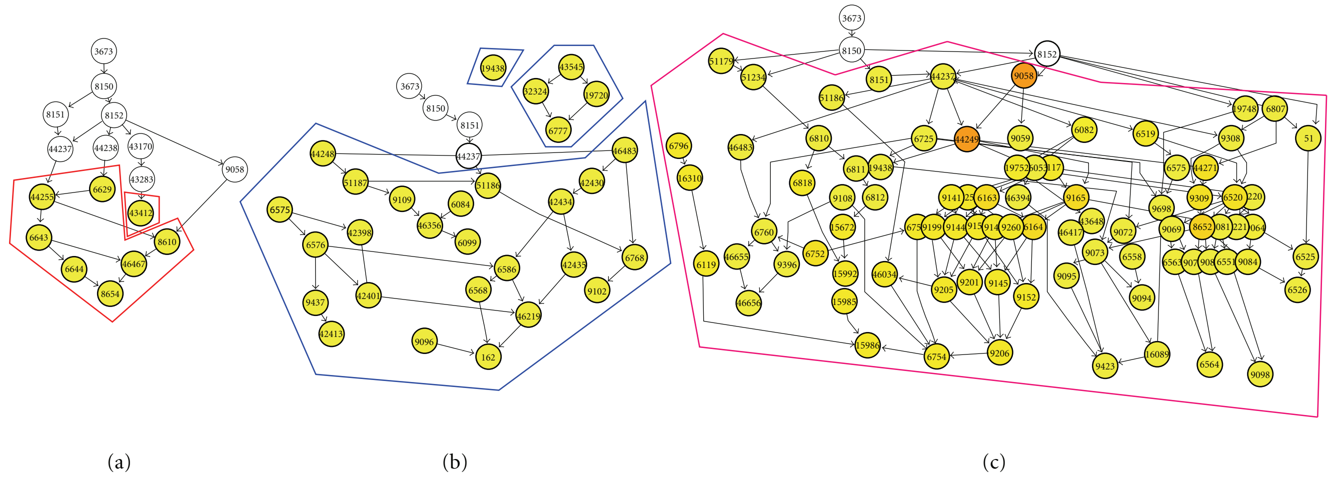

Figure 7: GO clusters resulting from V/SilR.

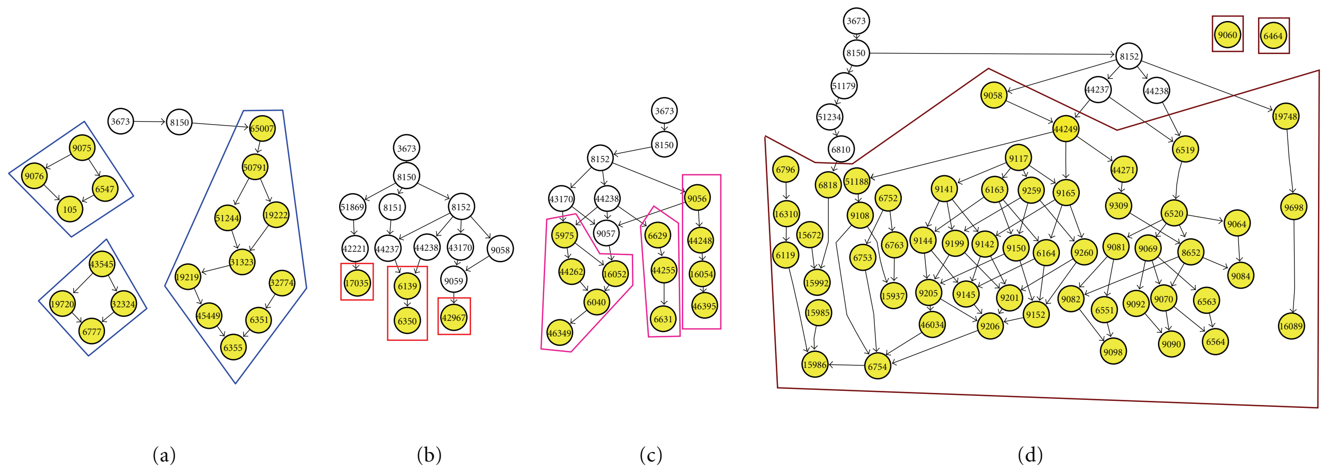

Figure 8: GO clusters resulting from V/SilT.

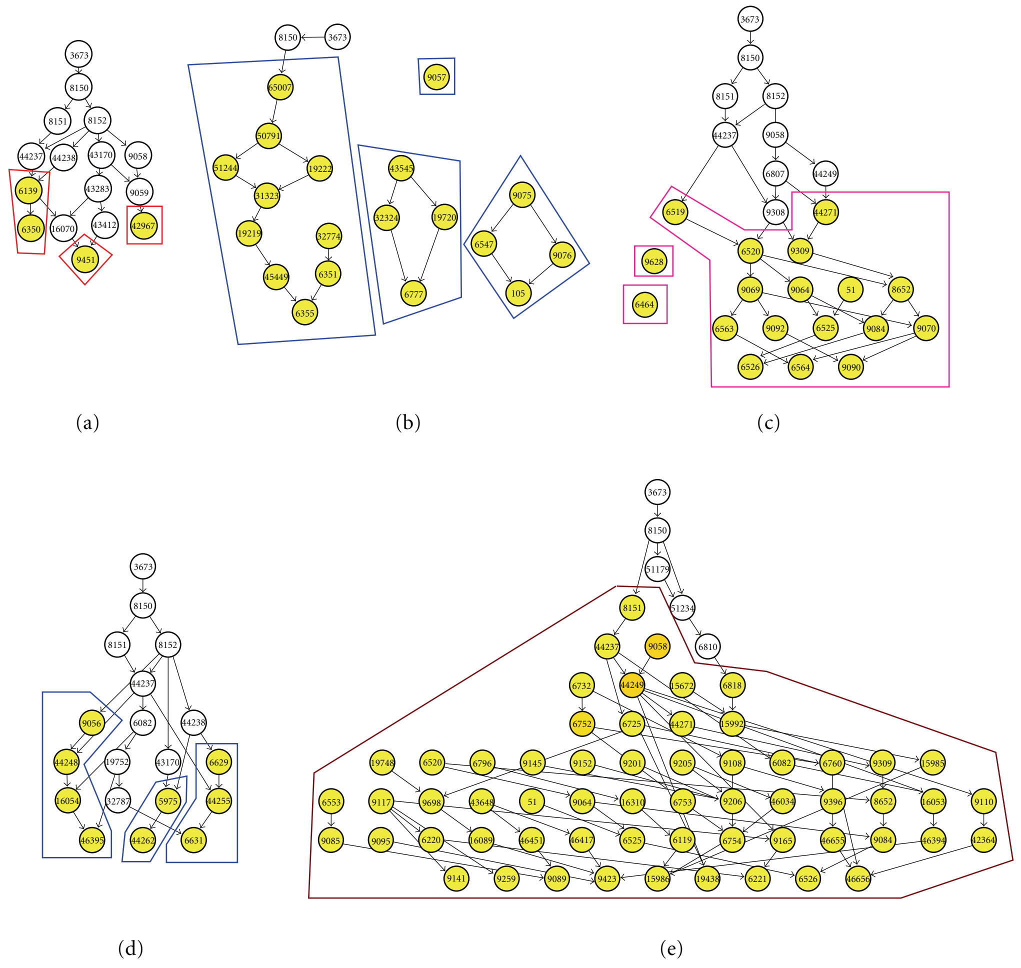

Figure 9: GO clusters resulting from V/BIC.

Table 5: Summary scores from *E. coli* data analysis.

| Cluster/algorithm | 1 | 2 | 3 | 4 | 5 | Total score | Average score |
|---|---|---|---|---|---|---|---|
| VSil/R | 153.14 | 2004.55 | 22129.80 | | | 24287.48 | 8095.83 |
| V/SilT | 405.73 | 3.10 | 82.95 | 7343.89 | | 7835.67 | 1958.92 |
| V/BIC | 4.42 | 422.42 | 513.70 | 44.64 | 11196.16 | 12181.33 | 2436.27 |

Given that algorithm was expected to group related functions together, the expectation for GO analysis was the creation of large, highly-connected subclusters within each main gene cluster. Ideally, one such subcluster would subsume the entire cluster; however, a small number of large subclusters within each cluster would validate the algorithm. The scoring equation (18) greatly rewards large, highly-connected subclusters; in fact, given a cluster, the score is maximized by having all GO terms within that cluster be connected within a single subcluster.

Figures 7, 8, and 9 show the results of the clustering using the three algorithms. Subclusters have been outlined for ease of identification. In some instances, nonenriched GO terms (colored white) have been removed for clarity. Visually, V/SilR is the better choice of the three. It has fewer overall clusters, and each cluster has generally fewer subclusters than V/SilT or V/BIC.

The clusters were scored using (18). Table 5 shows a summary of this analysis. As can be seen, V/SilR (3 clusters) far outscored both V/SilT (4 clusters) and V/BIC (5 clusters), both in aggregate and average cluster scores. Therefore, the conclusion is that V/SilR provides the better clustering performance.

## 4. Conclusion

Four combinations of VBEM algorithm and cluster statistics were tested. One of these, VBEM combined with the Silhouette statistic performed on the raw data clusters, clearly outperformed the other three in both simulated and real data tests. This method definitely shows promise in clustering time-series microarray data according to profile shape.

## Appendices

### A. Calculation of VBE Step

Let us assume we are on iteration $j+1$ and have both $q^{(j)}(C_g = k)$ and $q^{(j)}(\boldsymbol{\theta})$ available from iteration $j$. Then,

$$q^{(j+1)}(C_g = k) = \frac{\xi_g(k)}{\sum_{k=1}^{K} \xi_g(k)}, \tag{A.1}$$

where

$$\ln \xi_g(k) = \Psi(\gamma'_k) + \frac{1}{2}\sum_{n=1}^{N}\left[\left(\Psi\left(\frac{\alpha_{k,n}}{2}\right) - \ln\frac{\beta_{k,n}}{2}\right) - (\hat{\mu}_{k,n} - y_{g,n}) - \frac{N}{K_{k,n}}\right], \tag{A.2}$$

where $N$: number of time samples; $G$: number of genes (index $g$); $\Psi(\cdot)$: digamma function, and all other parameters are calculated from the VBM step.

### B. Calculation of VBM Step

Now we assume we have $q^{(j+1)}(C_g = k)$ from the prior VBE step. Then,

$$q^{(j+1)}(\mu_{n,k}, {\sigma_{n,k}}^2) = \mathrm{NIG}\left(\hat{\mu}_{n,k}, {\sigma_{n,k}}^2 \mid K_{k,n}, \frac{\alpha_{k,n}}{2}, \frac{\beta_{k,n}}{2}\right),$$
$$q^{j+1}(L) = \mathrm{Dir}(\dot{\gamma}_1, \dot{\gamma}_2, \ldots, \dot{\gamma}_k), \tag{B.1}$$

where $K_{k,n} = \sum_{g=1}^{G} q^j(C_g = k) + K$; $\hat{\mu}_{k,n} = K^{-1}{}_{k,n}[K_{\mu_{k,n,0}} + \sum_{g=1}^{G} q^{(j)}(C_g = k) y_{g,n}]$; $\alpha_{k,n} = \alpha_{k0} + \sum_{g=1}^{G} q^j(C_g = k)$; $\beta_{k,n} = \beta_{k0} + t(y)$; $t(\mathbf{y}) = \sum_{g=1}^{G}(C_g = k) {y_{g,n}}^2 + K_{{\mu_{k,n,0}}^2} + \hat{\mu}_{k,n}$; $\gamma_K{}' = \gamma_k + \sum_{g=1}^{G} q^{(j)}(C_g = k)$; NIG$(\cdot)$: Normal-Inverse-Gamma distribution; Dir$(\cdot)$: Dirichlet distribution.

### C. Calculation of Lower Bound $F(q(C_g), q(\boldsymbol{\theta}))$

Once $q^{(j+1)}(C_g = k)$ and $q^{(j+1)}(\theta)$ have been calculated, we calculate the lower bound using the following:

$$F(q(C_g), q(\boldsymbol{\theta})) = -\prod_{n=1}^{N}\prod_{k=1}^{K} \mathrm{KL}\left[\frac{q(\mu_{n,k}, {\sigma_{n,k}}^2)}{p(\mu_{n,k}, {\sigma_{n,k}}^2)}\right] - \mathrm{KL}\left[\frac{q(\mathbf{L})}{p(\mathbf{L})}\right] + \ln Z, \tag{C.1}$$

$$\begin{aligned}\mathrm{KL}\left[\frac{q(\mu_{n,k}, {\sigma_{n,k}}^2)}{p(\mu_{n,k}, {\sigma_{n,k}}^2)}\right] &= -\frac{1}{2}\left(\ln\frac{K}{K_{n,k}} - \frac{K}{2K_{n,k}} + 1\right) + \frac{\alpha_{k,n}}{2}\left(\Psi\left(\frac{\alpha_{k,n}}{2}\right) + \frac{\beta_0}{\beta_{k,n}} - 1\right)\\ &\quad + \cdots + \frac{\alpha_0}{2}\left(\ln\frac{\beta_{k,n}}{2} - \Psi\left(\frac{\alpha_{k,n}}{2}\right) - \ln\frac{\beta_0}{2}\right)\\ &\quad + \frac{K}{2}(\hat{\mu}_{n,k} - \mu_0)^2\frac{\alpha_{k,n}}{\beta_{k,n}} - \ln\frac{\Gamma(\alpha_{k,n}/2)}{\Gamma(\alpha_0/2)},\end{aligned} \tag{C.2}$$

$$\mathrm{KL}\left[\frac{q(\mathbf{L})}{p(\mathbf{L})}\right] = \ln\frac{\Gamma(\gamma'_0)}{\Gamma(\gamma_0)} - \sum_{k=1}^{K}\left[\ln\frac{\Gamma(\gamma'_k)}{\Gamma(\gamma_k)} - (\gamma'_k - \gamma_k)(\Psi(\gamma'_k) - \Psi(\gamma'_0))\right], \tag{C.3}$$

$$\ln Z = \sum_{g=1}^{G} \ln Z_g, \tag{C.4}$$

where $Z_g = \xi\sum_{k=1}^{K}\xi_g(k)$ and $\ln\xi = -(N/2)\ln 2\pi - \Psi(\gamma'_0)$.

## Acknowledgment

This work is supported in part by NSF Grant CCF-0546345. Dr. Tim Lilburn has been instrumental with his assistance and guidance.

## References

[1] D. Jiang, C. Tang, and A. Zhang, "Cluster analysis for gene expression data: a survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 11, pp. 1370–1386, 2004.

[2] M. H. Asyali, D. Colak, O. Demirkaya, and M. S. Inan, "Gene expression profile classification: a review," *Current Bioinformatics*, vol. 1, no. 1, pp. 55–73, 2006.

[3] Z. Bar-Joseph, G. K. Gerber, D. K. Gifford, T. S. Jaakkola, and I. Simon, "Continuous representations of time-series gene expression data," *Journal of Computational Biology*, vol. 10, no. 3-4, pp. 341–356, 2003.

[4] 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 Research*, vol. 34, no. 4, pp. 1261–1269, 2006.

[5] L. Rueda, A. Bari, and A. Ngom, "Clustering time-series gene expression data with unequal time intervals," in *Transactions on Computational Systems Biology X*, vol. 5410 of *Lecture Notes in Computer Science*, pp. 100–123, Springer, Berlin, Germany, 2008.

[6] Y. Yuan and C.-T. Li, "Unsupervised clustering of gene expression time series with conditional random fields," in *Proceedings of the Inaugural IEEE International Conference on Digital EcoSystems and Technologies (DEST '07)*, pp. 571–576, Cairns, Australia, February 2007.

[7] K. Kim, S. Zhang, K. Jiang, et al., "Measuring similarities between gene expression profiles through new data transformations," *BMC Bioinformatics*, vol. 8, article 29, pp. 1–14, 2007.

[8] X. Wen, S. Fuhrman, G. S. Michaels, et al., "Large-scale temporal gene expression mapping of central nervous system development," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 95, no. 1, pp. 334–339, 1998.

[9] C. S. Möller-Levet, F. Klawonn, K.-H. Cho, H. Yin, and O. Wolkenhauer, "Clustering of unevenly sampled gene expression time-series data," *Fuzzy Sets and Systems*, vol. 152, no. 1, pp. 49–66, 2005.

[10] R. Balasubramaniyan, E. Hüllermeier, N. Weskamp, and J. Kämper, "Clustering of gene expression data using a local shape-based similarity measure," *Bioinformatics*, vol. 21, no. 7, pp. 1069–1077, 2005.

[11] 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," in *Proceedings of the 8th Pacific Symposium on Biocomputing (PSB '03)*, pp. 351–362, Lihue, Hawaii, USA, January 2003.

[12] B. Tjaden, "An approach for clustering gene expression data with error information," *BMC Bioinformatics*, vol. 7, article 17, pp. 1–15, 2006.

[13] A. Ben-Hur, A. Elisseeff, and I. Guyon, "A stability based method for discovering structure in clustered data," in *Proceedings of the 7th Pacific Symposium on Biocomputing (PSB '02)*, pp. 6–17, Lihue, Hawaii, USA, January 2002.

[14] E. Dimitriadou, S. Dolničar, and A. Weingessel, "An examination of indexes for determining the number of clusters in binary data sets," *Psychometrika*, vol. 67, no. 1, pp. 137–159, 2002.

[15] S. Dudoit and J. Fridlyand, "A prediction-based resampling method for estimating the number of clusters in a dataset," *Genome Biology*, vol. 3, no. 7, pp. 1–21, 2002.

[16] R. Tibshirani, G. Walther, and T. Hastie, "Estimating the number of clusters in a data set via the gap statistic," *Journal of the Royal Statistical Society. Series B*, vol. 63, no. 2, pp. 411–423, 2001.

[17] H. Sun and M. Sun, "Trail-and-error approach for determining the number of clusters," in *Proceedings of the 4th International Conference on Machine Learning and Cybernetics (ICMLC '05)*, vol. 3930 of *Lecture Notes in Computer Science*, pp. 229–238, Guangzhou, China, August 2006.

[18] D. L. Wild, C. E. Rasmussen, and Z. Ghahramani, "A Bayesian approach to modeling uncertainty in gene expression clusters," in *Proceedings of the 3rd International Conference on Systems Biology (ICSB '02)*, Stockholm, Sweden, December 2002.

[19] Y. Xu, V. Olman, and D. Xu, "Minimum spanning trees for gene expression data clustering," *Genome Informatics*, vol. 12, pp. 24–33, 2001.

[20] M. Yan and K. Ye, "Determining the number of clusters using the weighted gap statistic," *Biometrics*, vol. 63, no. 4, pp. 1031–1037, 2007.

[21] M. J. Beal and Z. Ghahramani, "The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures," in *Proceedings of the 7th Valencia International Meeting on Bayesian Statistics*, vol. 7, pp. 453–464, Tenerife, Spain, June 2003.

[22] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987.

[23] G. Schwarz, "Estimating the dimension of a model," *Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[24] The University of Oklahoma's *E. coli* Gene Expression Database, http://chase.ou.edu/oubcf/.

[25] The Entrez Genome Database. National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, "Escherichia coli K-12 data," http://www.ncbi.nlm.nih.gov/.

[26] P. Shannon, A. Markiel, O. Ozier, et al., "Cytoscape: a software environment for integrated models of biomolecular interaction networks," *Genome Research*, vol. 13, no. 11, pp. 2498–2504, 2003.

[27] S. Maere, K. Heymans, and M. Kuiper, "*BiNGO*: a Cytoscape plugin to assess overrepresentation of gene ontology categories in biological networks," *Bioinformatics*, vol. 21, no. 16, pp. 3448–3449, 2005.

[28] Y. Yuan and C.-T. Li, "Probabilistic framework for gene expression clustering validation based on gene ontology and graph theory," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '08)*, pp. 625–628, Las Vegas, Nev, USA, March-April 2008.