Journal of Bioinformatics and Computational Biology
Vol. 7, No. 2 (2009) 339–356

Imperial College Press
www.icpress.co.uk

SIMULTANEOUSLY SEGMENTING MULTIPLE GENE EXPRESSION TIME COURSES BY ANALYZING CLUSTER DYNAMICS

SATISH TADEPALLI*, NAREN RAMAKRISHNAN†
and LAYNE T. WATSON‡

Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061, USA
**stadepal@cs.vt.edu*
†*naren@cs.vt.edu*
‡*ltw@cs.vt.edu*

BUD MISHRA

Courant Institute of Mathematical Sciences
New York University
New York, NY 10012, USA
mishra@nyu.edu

RICHARD F. HELM

Department of Biochemistry
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061, USA
helmrf@vt.edu

Received 2 June 2008
Revised 18 November 2008
Accepted 16 December 2008

We present a new approach to segmenting multiple time series by analyzing the dynamics of cluster formation and rearrangement around putative segment boundaries. This approach finds application in distilling large numbers of gene expression profiles into temporal relationships underlying biological processes. By directly minimizing information-theoretic measures of segmentation quality derived from Kullback-Leibler (KL) divergences, our formulation reveals clusters of genes along with a segmentation such that clusters show concerted behavior within segments but exhibit significant regrouping across segmentation boundaries. The results of the segmentation algorithm can be summarized as Gantt charts revealing temporal dependencies in the ordering of key biological processes. Applications to the yeast metabolic cycle and the yeast cell cycle are described.

Keywords: Time series segmentation; clustering; KL-divergence; temporal regulation.

1. Introduction

Time course analysis has become an important tool for the study of developmental, disease progression, and cyclical biological processes, e.g. the cell cycle,[1] metabolic cycle,[2] and even entire life cycles.[3] When the number of time points is large, researchers have studied using continuous representations to smooth out noise,[4] using the application of hidden Markov models to guide clustering,[5] and using static measurements to "fill in the gaps" in the time series data.[6] When the number of time points is small, researchers have studied the role played by sampling rates[7] and proposed the use of model profiles[8] to guide clustering. Recently, researchers are quantifying timing differences in gene expression,[9] and also reconstructing regulatory relationships.[10]

One of the attractions of time series analysis is its promise to reveal temporal relationships underlying biological processes: which process occurs before what, what are the "checkpoints" that must be satisfied (and when), and whether there can be alternative pathways of time series progression. Although such analysis can be conducted by tracking individual genes whose function is known, we desire to automatically mine, in an unsupervised manner, temporal relationships involving *groups* of genes, which are not *a priori* defined. In particular, we desire to identify both segments of the time course where groups show concerted behavior and boundaries between segments where there is significant "regrouping" of genes. We cast this problem as a form of time series segmentation where the segmentation criterion is driven by measures over cluster dynamics.

2. Preview of Results

It is important to contrast our goals with prior work. Typical works on time series segmentation are focused on segmenting a single time series and their goal is to partition the dataset into internally homogeneous segments. Variations of dynamic programming[11] and Bayesian approaches[12] have been used to solve this problem. When multiple time series are involved, it is assumed that all the series have a similar pattern in a given segment. Algorithms based on fuzzy clustering[13] and graphical models[14] have been applied in this context. Essentially, all these works are based on homogeneity assumptions within segments and model the segmentation problem as clustering time points with the constraint that data samples in a cluster must belong to successive time points. Algorithms to mine the temporal order of events occurring in multiple time series have also been under investigation. Moerchen *et al.*[15] devised a temporal grammar for this purpose. However, their approach requires manual partitioning of the time series, and the events are derived by naive discretization of the multiple time series. We describe an approach to segment multiple time series without making any homogeneity assumptions and automatically mine the temporal sequence of events occurring in the data. We explicitly model each segment as a heterogeneous mix of multiple clusters which can themselves be redefined across segments. Our work is hence directly targeted

to mining datasets involving thousands of genes where there are complex inter-relationships and re-organizations underlying the dataset.

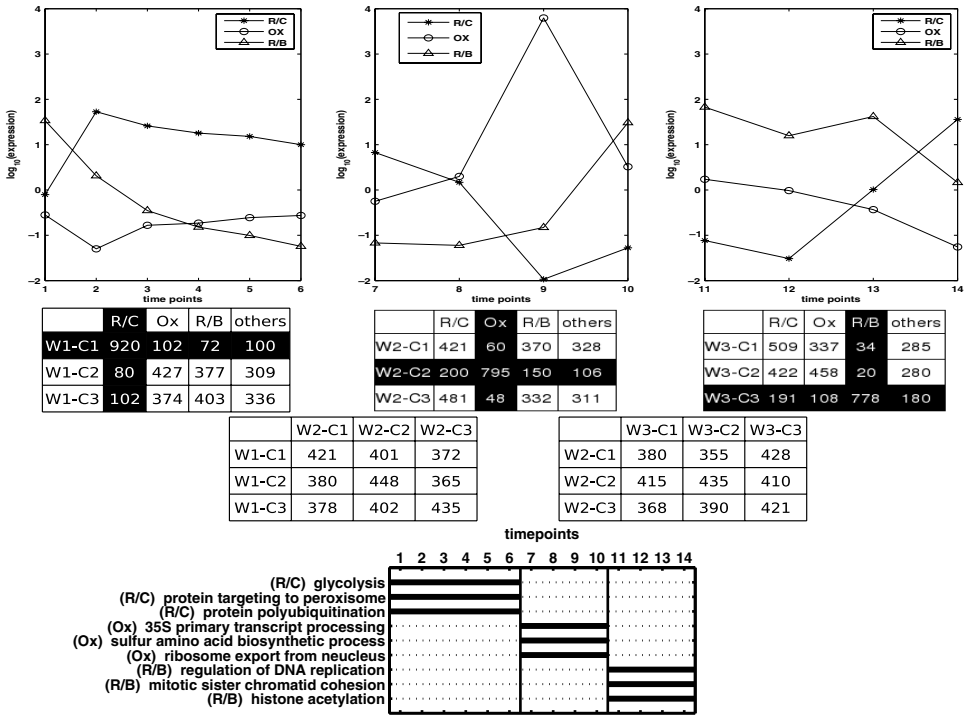
As an example, consider the Yeast metabolic cycle (YMC), using the dataset of Tu *et al.*[2] The YMC is a carefully coordinated mechanism between a reductive charging (R/C) phase involving non-respiratory metabolism (glycolysis, fatty acid oxidation) and protein degradation, followed by oxidative metabolism (Ox), where respiratory processes are used to generate adenosine triphosphate (ATP), culminating in reductive metabolism (R/B), characterized by a decrease in oxygen uptake and emphasis on DNA replication, mitochondrial biogenesis, and cell division. Different genes are central to each of these phases. Tu *et al.*[2] analyzed this 36-pt time course — spanning approximately three cycles (R/B phase is not sampled in the last cycle) — by tracking "sentinel" genes showing periodic behavior across the time course.

We analyzed this dataset of 3602 gene expression profiles over a 15 h period using our segmentation algorithm and arrived at a segmentation corresponding to three cycles. For lack of space, we present only one of these cycles, as shown in Fig. 1. Figure 1 (top) displays *what we found*, Fig. 1 (second and third row) display *how we found it*, and Fig. 1 (bottom row) displays *why what we found is meaningful.* It is important to note that the only information supplied to the segmentation algorithm is the set of time course profiles of all genes, without any supervised information about their membership in different categories.

First, the segmentation we determined using our algorithm is a breakdown of the timepoints into [1–6], [7–10], and [11–14]. To understand this segmentation, we plot the mean profile of genes in each category *a posteriori* in Fig. 1 (top) indicating that each group of genes characteristically peaks in one segment. Thus, although our algorithm was not given the functional membership of genes, the segmentation brings out the aspect that different groups are active in different segments.

To see how our algorithm identifies the segmentation, consider the second and third rows of Fig. 1. We assume a setting of three clusters so that in every segment identified by the algorithm, the genes within the segment are partitioned into three clusters. The second row displays contingency tables where the clusters identified by our algorithm are the rows, pre-defined functional categories are the columns, and the cells show the overlap between the clusters and categories. We see that, indeed, there is one cluster in each segment that brings together the relevant functional set of genes automatically. Specifically, the cluster W1-C1 (i.e. cluster 1 of window 1) has high overlap with the functional category R/C whereas the other clusters and categories do not show any significant overlap. Similarly, cluster W2-C2 shows high overlap with the Ox functional category, as does W3-C3 with R/B.

The actual criterion used by our segmentation algorithm is shown in Fig. 1 (third row) that indicates that the overlap between clusterings across segment boundaries is highly dissimilar. This validates our hypothesis that there are characteristic groups of genes within each segment and they re-organize around the segment boundaries.



	R/C	Ox	R/B	others
W1-C1	920	102	72	100
W1-C2	80	427	377	309
W1-C3	102	374	403	336

	R/C	Ox	R/B	others
W2-C1	421	60	370	328
W2-C2	200	795	150	106
W2-C3	481	48	332	311

	R/C	Ox	R/B	others
W3-C1	509	337	34	285
W3-C2	422	458	20	280
W3-C3	191	108	778	180

	W2-C1	W2-C2	W2-C3
W1-C1	421	401	372
W1-C2	380	448	365
W1-C3	378	402	435

	W3-C1	W3-C2	W3-C3
W2-C1	380	355	428
W2-C2	415	435	410
W2-C3	368	390	421

Fig. 1. Preview of results from segmenting the Yeast metabolic cycle (YMC) time series dataset. Only one cycle is shown here. The YMC involves the staged coordination of a reductive, charging phase (R/C, time points [1–6]), followed by oxidative metabolism (Ox, time points [7–10]), followed by reductive metabolism (R/B, time points [11–14]). (top row) Mean expression profiles of certain groups of genes peak during each phase and these genes are assigned to those phases. (second row) Genes assigned to the particular phase are heavily clustered together during each segment as indicated by the highlighted row and column; e.g. majority of R/C genes are clustered together in cluster 1 of segment 1 (denoted by W1-C1) as shown in the left most contingency table (third row). Contingency tables with rows representing clusters in the segment to the left and columns representing the clusters in the segment to the right capture the dispersion in the clusters across the segments. The first row in the left most table which represents how the genes clustered together in cluster 1 of segment 1 (W1-C1) are now spread into the clusters in segment 2 (W2-C1, W2-C2, and W3-C3) (fourth row). Key biological processes enriched in each segment (only a few are shown due to lack of space).

A final view of the segmentation is given in Fig. 1 (bottom row) where we conduct a functional enrichment analysis and display the categories most significantly enriched in each segment (in some cluster). Only a few enriched categories are shown here, for ease of illustration. Thus, the time-bounded enrichments lead to a Gantt chart view of the YMC which identifies biological processes with modulated activities in different segments.

We reiterate that the time point boundaries, the groups of genes important in each segment, and the functions enriched in them, are inferred automatically.

No explicit modeling of periodicity or other prior biological knowledge has been imparted to the segmentation algorithm.

3. Problem Formulation

Our working hypothesis is that genes function in groups and that such groupings are dynamically redefined at important stages of time series progression. Hence identifying the groups as well as the points at which the regroupings happen is critical. We refer to this as a cluster dynamics approach since we are modeling movement of genes into and out of clusters and studying relationships between clusters from neighboring segments.

We are given multiple vectors of measurements $\mathcal{G} = \{\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_N\}$, where each $\mathbf{g}_i$ is a time series over $\mathcal{T} = \{t_1, t_2, \ldots, t_l\}$. The problem of segmentation is to express $\mathcal{T}$ as a sequence of segments or windows: $(w_{t_1}^{t_a}, w_{t_{a+1}}^{t_b}, \ldots, w_{t_k}^{t_l})$ where each window $w_{t_s}^{t_e}$, $t_s \leq t_e$, is a set of consecutive time points beginning at (and inclusive of) time point t_s and ending at (and inclusive of) time point t_e.

We first describe a way to evaluate a given segmentation before presenting an algorithm for identifying segmentations. We begin by studying the case of just two adjacent windows: $w_{t_a}^{t_b}$ and $w_{t_{b+1}}^{t_c}$. Given two clusterings of genes, one for each of the windows, our evaluation criterion requires that these two sets of clusters are highly dissimilar, i.e. genes clustered together in some cluster of $w_{t_a}^{t_b}$ move out of their clusters and are clustered together with different genes in $w_{t_{b+1}}^{t_c}$. As an example, consider the following contingency tables from a dataset involving 18 genes and three clusters in each window.

2	2	2
2	2	2
2	2	2

(a)

6	0	0
0	6	0
0	0	6

(b)

0	6	0
0	0	6
6	0	0

(c)

Here the rows refer to clusters of $w_{t_a}^{t_b}$ and the columns refer to clusters of $w_{t_{b+1}}^{t_c}$ and the cells represent the overlaps in the clusters across the windows. The table (a) represents the case in which the clusters across the windows have maximum dissimilarity: each cluster in $w_{t_{b+1}}^{t_c}$ is comprised of genes taken from all the clusters in $w_{t_a}^{t_b}$ and *vice versa*. The tables (b) and (c) represent the cases where the clusters in both windows have high similarity: the same sets of genes are clustered together across the windows (resulting in overlap counts of 6). We require that our criterion favors clusters as shown in table (a) to those in tables (b) and (c). Each row and column of this contingency table can be interpreted as a normalized probability distribution. In the case of table (a) each distribution is $[1/3, 1/3, 1/3]$, which is a uniform distribution while in the case of tables (b) and (c), each distribution is $[0, 1, 0]$ which has a maximum deviation from the uniform distribution. We use this observation to formulate our criterion as described below.

Formally, given two windows $w_{t_a}^{t_b}$ and $w_{t_{b+1}}^{t_c}$, we seek r clusters in the window $w_{t_a}^{t_b}$ and c clusters in the window $w_{t_{b+1}}^{t_c}$. Let $\alpha = \{1, \ldots, r\}$ and $\beta = \{1, \ldots, c\}$ represent the cluster random variables for the windows $w_{t_a}^{t_b}$ and $w_{t_{b+1}}^{t_c}$, respectively. The similarity of the clusters in the windows is measured using a two dimensional contingency table. The number n_{ij} in the (i, j)th cell of the contingency table represents the number of samples that are clustered together in cluster i of window $w_{t_a}^{t_b}$ and cluster j of window $w_{t_{b+1}}^{t_c}$. The sizes of the clusters in the window $w_{t_a}^{t_b}$ are obtained by the column-wise sums across each row ($n_{i\cdot} = \sum_{j=1}^{c} n_{ij}$), while the sizes of clusters in the window $w_{t_{b+1}}^{t_c}$ are obtained by the row-wise sums down each column ($n_{\cdot j} = \sum_{i=1}^{r} n_{ij}$). Also note that, N, the total number of data samples in each window, is obtained by the sum of all cluster sizes in the window: $\sum_{i=1}^{r} n_{i\cdot} = \sum_{j=1}^{c} n_{\cdot j} = N$. Given these counts, the normalized contingency table of cluster similarity counts can be interpreted as a joint distribution of the cluster variables α and β,

$$P(\alpha = i, \beta = j) = \frac{n_{ij}}{N}, \quad i = 1, \ldots, r, \quad \text{and} \quad j = 1, \ldots, c. \tag{1}$$

The row-wise distributions of this contingency table represent the conditional probabilities of the clusters in window $w_{t_{b+1}}^{t_c}$ given the clusters in window $w_{t_a}^{t_b}$. Similarly, the column-wise distributions represent the probabilities of the clusters in window $w_{t_a}^{t_b}$ given the clusters in window $w_{t_{b+1}}^{t_c}$. We define r random variables $\{R_1, \ldots, R_r\}$ corresponding to the r rows, and c random variables $\{C_1, \ldots, C_c\}$ corresponding to the c columns. The probabilities of these random variables are then defined in terms of the conditional probabilities of the cluster variables α and β as follows:

$$P(R_i = j) = P(\beta = j | \alpha = i) = \frac{P(\alpha = i, \beta = j)}{P(\alpha = i)} = \frac{n_{ij}}{n_{i\cdot}}, \tag{2}$$

$$P(C_j = i) = P(\alpha = i | \beta = j) = \frac{P(\alpha = i, \beta = j)}{P(\beta = j)} = \frac{n_{ij}}{n_{\cdot j}}. \tag{3}$$

Each row variable R_i takes c values from the columns corresponding to the ith row as given by the probability mass function p_{R_i}, and similarly each column variable C_j takes r values from the rows corresponding to jth column as given by the probability mass function p_{C_j}. Since each row has c cells, the uniform distribution over the cells in each row is $U(\frac{1}{c})$, and for each column the uniform distribution is $U(\frac{1}{r})$. We capture the deviation of these row-wise and column-wise distributions w.r.t. the uniform distribution as:

$$\mathcal{F} = \frac{1}{r} \sum_{i=1}^{r} D_{KL}\left(p_{R_i} \| U(\tfrac{1}{c})\right) + \frac{1}{c} \sum_{j=1}^{c} D_{KL}\left(p_{C_j} \| U(\tfrac{1}{r})\right), \tag{4}$$

where $D_{KL}(p\|q) = \sum_x p(x) \log_2 \frac{p(x)}{q(x)}$ is the Kullback-Leibler (KL) divergence between two probability distributions with probability mass functions $p(x)$ and $q(x)$, and $U(\cdot)$ denotes the uniform distribution whose argument is the probability of any outcome.

Thus, we have r KL-divergences, one for each row, and c KL-divergences, one for each column. In order to mitigate the effect of lopsided contingency tables (where either $r \gg c$ or $c \gg r$), note that the sums of these divergences are in turn averaged (row-wise and column-wise). In such cases, it is possible to optimize $\mathcal{F}$ by focusing on the "longer" dimension without really ensuring that the other dimension's projections are close to uniform. Finally, note that Eq. (4) can be readily extended to the case where we have more than two segments.

The objective function defined in Eq. (4) has connections to the principle of minimum discrimination information (MDI) introduced by Kullback for the analysis of contingency tables.[16] The MDI principle states that if q is the assumed or true distribution, the estimated distribution p must be chosen such that $D_{KL}(p\|q)$ is minimized. In our case, q is the uniform distribution desired and p is the distribution estimated from observed data.

By definition of KL-divergence, the objective function in Eq. (4) can be expressed as

$$\begin{aligned}\mathcal{F} &= -\frac{1}{r}\sum_{i=1}^{r} H(R_i) + \log_2(c) - \frac{1}{c}\sum_{j=1}^{c} H(C_j) + \log_2(r) \\ &= -\frac{1}{r}\sum_{i=1}^{r} H(\beta|\alpha = i) - \frac{1}{c}\sum_{j=1}^{c} H(\alpha|\beta = j) + \log_2(r \cdot c), \end{aligned} \tag{5}$$

where $H(X)$ is the entropy of the random variable X with probability mass function $p(x)$ and is defined as $H(X) = -\sum_x p(x)\log_2(p(x))$. Entropy is a measure of the uncertainty of a random variable. Minimizing $\mathcal{F}$ leads to a high entropy of the cluster conditional distributions $P(\beta|\alpha = i)$ and $P(\alpha|\beta = j)$. Intuitively, this means that given a cluster in window $w_{t_a}^{t_b}$, it is difficult to predict a cluster in the adjacent window $w_{t_{b+1}}^{t_c}$ and *vice versa*. This occurs when the clusters in the adjacent windows are highly dissimilar as required by the independent clusters we wish to find in the adjacent windows. On the other hand, maximizing $\mathcal{F}$ leads to a low entropy of the cluster conditional distributions and thus maximally similar clusters across the windows (this aspect is not studied further in this paper). However, for ease of interpretation and connections to the MDI principle, we prefer to represent the objective function $\mathcal{F}$ in terms of KL-divergence as shown in Eq. (4).

Minimizing $\mathcal{F}$ will yield row-wise and column-wise distribution estimates that are close to the respective uniform distributions and, hence, result in independent clusterings across the neighboring windows.

4. Clustering Across Windows

We now turn our attention to the clustering algorithm that must balance two conflicting criteria: namely, the clusters across neighboring windows must be independent and, yet the clusters must exhibit concerted behavior within a window.

The former criterion is modeled as described previously while the latter is achieved by parameterizing the clusters using soft cluster prototypes as described below.

4.1. *Parameterizing cluster prototypes*

We develop our notation for two adjacent windows, and the extension to greater numbers of windows is straightforward. Given a gene vector $\mathbf{g}_k$, let its projection onto the "left" window $w_{t_a}^{t_b}$ be referred to as $\mathbf{x}_k$, and its projection onto the "right" window $w_{t_{b+1}}^{t_c}$ be referred to as $\mathbf{y}_k$. Recall that sets of such projections are clustered separately such that the clusters are maximally dissimilar. Let r and c be the number of clusters for $\mathbf{x}$ and $\mathbf{y}$ vectors, which results in an $r \times c$ contingency table. Let $\mathbf{m}_i^{(x)}$ be the prototype vector for the ith cluster of the $\mathbf{x}$ vectors. The random variable $V^{(\mathbf{x}_k)}$ denotes the assignment of the data vector $\mathbf{x}_k$ to the clusters: the probability of $\mathbf{x}_k$ being assigned to cluster i is given by $P(V^{(\mathbf{x}_k)} = i) = v_i^{(\mathbf{x}_k)}$, where $\sum_{i=1}^{r} v_i^{(\mathbf{x}_k)} = 1$. We refer to the probabilities $v_i^{(\mathbf{x}_k)}$ as cluster membership indicator variables. Similar cluster prototypes $\mathbf{m}_j^{(y)}$, random variables $V^{(\mathbf{y}_k)}$, and cluster indicator variables $v_j^{(\mathbf{y}_k)}$ are defined for $\mathbf{y}$ vectors as well. We denote the probability mass function associated with each $V^{(\mathbf{x}_k)}$ as $p_{V^{(\mathbf{x}_k)}}$, and with each $V^{(\mathbf{y}_k)}$ as $p_{V^{(\mathbf{y}_k)}}$. Then the contingency table counts can be calculated as $n_{ij} = \sum_{k=1}^{N} v_i^{(\mathbf{x}_k)} v_j^{(\mathbf{y}_k)}$. In *hard clustering* algorithms, like the traditional k-means, each data sample is assigned to the nearest cluster with a probability of 1. However, calculating n_{ij} using hard memberships renders the function $\mathcal{F}$ in Eq. (4) nondifferentiable at certain points, as a result of which, we cannot leverage classical numerical optimization algorithms to minimize $\mathcal{F}$. To avoid this problem, cluster indicator variables are typically treated as continuous real variables making $\mathcal{F}$ a smooth function that is continuously differentiable and assigning a nonzero cluster membership probability for each data sample, i.e. $v_i^{(\mathbf{x}_k)}, v_j^{(\mathbf{y}_k)} \in (0, 1)$.

There are many ways of smoothing, one approach being the use of a Gaussian kernel between the vector and the cluster prototype. We present a novel derivation of this kernel that explains how the error in the smoothing can be explicitly controlled and also suggests other formulations for smoothing. First, we define

$$\gamma_{(i,i')}(\mathbf{x}_k) = \frac{\|\mathbf{x}_k - \mathbf{m}_{i'}^{(x)}\|^2 - \|\mathbf{x}_k - \mathbf{m}_i^{(x)}\|^2}{D}, 1 \le i, i' \le r, \tag{6}$$

where $D = \max_{k,k'} \|\mathbf{x}_k - \mathbf{x}_{k'}\|^2, 1 \le k, k' \le N$ is the pointset diameter.

The non-normalized cluster assignment probabilities are given by

$$\hat{v}_i^{\mathbf{x}_k} = \exp\left(\rho\left(\min_{i'} \gamma_{(i,i')}(\mathbf{x}_k)\right)\right), \tag{7}$$

and the normalized probabilities are then given by

$$v_i^{(\mathbf{x}_k)} = \frac{\hat{v}_i^{\mathbf{x}_k}}{\sum_{i'} \hat{v}_{i'}^{\mathbf{x}_k}}. \tag{8}$$

A well known approximation to $\min_{i'} \gamma_{(i,i')}(\mathbf{x}_k)$ is the Kreisselmeier-Steinhauser (KS) envelope function[17,18] given by

$$KS_i(\mathbf{x}_k) = \frac{-1}{\rho} \ln \left[\sum_{i'=1}^{r} \exp(-\rho \gamma_{(i,i')}(\mathbf{x}_k)) \right], \tag{9}$$

where $\rho \gg 0$. The KS function is a smooth function that is infinitely differentiable. Using this, the cluster membership indicators are redefined as:

$$v_i^{(\mathbf{x}_k)} = \frac{\exp[\rho KS_i(\mathbf{x}_k)]}{\sum_{i'=1}^{r} \exp[\rho KS_{i'}(\mathbf{x}_k)]} = \frac{\exp(-\frac{\rho}{D}\|x_k - m_i\|^2)}{\sum_{i'=1}^{r} \exp(-\frac{\rho}{D}\|x_k - m_i'\|^2)}. \tag{10}$$

The cluster memberships for the "right" window, $v_j^{(\mathbf{y}_k)}$, are also smoothed similarly.

The astute reader would have noticed that ρ/D is the width of the Gaussian kernel used for approximation but the KS-function helps tease out how the width must be set in order to achieve a certain quality of approximation. Notice that D is completely determined by the data but ρ is a user-settable parameter, and precisely what we can tune. The KS-function provides bounds on the error with which it approximates $\min_{i'} \gamma_{(i,i')}(\mathbf{x}_k)$, given as follows:

$$\min_{i'} \gamma_{(i,i')}(\mathbf{x}_k) - \frac{\ln(N)}{\rho} \leq KS_i(\mathbf{x}_k) \leq \min_{i'} \gamma_{(i,i')}(\mathbf{x}_k). \tag{11}$$

The above inequality shows that the value of ρ can be determined for a given error precision. Furthermore, we can use any other rapidly decaying function to assign the probabilities (i.e. in place of the exponential) and the KS-function would again help us smooth the resulting assignments, but will yield assignment probabilities that are quite different from the traditional Gaussian kernel. In this sense, the KS-approximation is a versatile approach to smooth a variety of functions.

4.2. *Regularized objective function for independent clusters*

Minimizing the function $\mathcal{F}$ in Eq. (4) should ideally yield clusters that are independent across windows and local within each window. However, using smooth cluster prototypes gives rise to an alternative minimum solution where each data sample is assigned with uniform probability to each cluster. For example, recall the uniform contingency table example, in Sec. 3. Each of the 18 samples can be assigned to the three row clusters and three column clusters with probability $[1/3, 1/3, 1/3]$ and the estimate of the count matrix from these soft counts would still be uniform in each cell ($\sum_k v_i^{(\mathbf{x}_k)} v_j^{(\mathbf{y}_k)} = 2$). To avoid degenerate solutions such as these, we require maximum deviation of individual data vector probabilities ($v_i^{(\mathbf{x}_k)}$ and $v_j^{(\mathbf{y}_k)}$) from the uniform distribution over the number of clusters. This leads to the regularized

objective function:

$$\mathcal{F} = \frac{\lambda}{r}\sum_{i=1}^{r} D_{KL}\left(p_{R_i}\|U(\tfrac{1}{c})\right) + \frac{\lambda}{c}\sum_{j=1}^{c} D_{KL}\left(p_{C_j}\|U(\tfrac{1}{r})\right) - \frac{1}{N}\sum_{k=1}^{N} D_{KL}\left(p_{V^{(\mathbf{x}_k)}}\|U(\tfrac{1}{r})\right) - \frac{1}{N}\sum_{k=1}^{N} D_{KL}\left(p_{V^{(\mathbf{y}_k)}}\|U(\tfrac{1}{c})\right), \quad (12)$$

where λ is the weight, set to a value greater than 1, to give more emphasis to minimizing the row and column distributions. This also enforces equal cluster sizes. The role of λ is to enforce a "balancing constraint" on the clusters (i.e. approximately equal cluster sizes) and to prevent samples from being assigned to multiple clusters. Other works have focused on explicitly capturing these aspects by an objective function (e.g. see Ref. 19) but here we intend λ to be a regularization parameter, to avoid degenerate solutions. Hence the exact value of λ is not as crucial as the regime in which we conduct the optimization. In order to adjust λ, we vary its value over a range (typically [1,2] in small step sizes). Based on experimentation, the cluster assignments of most gene vectors (more than 90%) do not change after a particular value of λ and we use this criterion to set λ. All the terms in Eq. (12) above can be calculated in terms of the smoothed cluster membership probabilities $v_i^{(\mathbf{x}_k)}$ and $v_j^{(\mathbf{y}_k)}$, which are in turn calculated in terms of the cluster prototypes $\mathbf{m}_i^{(x)}$ and $\mathbf{m}_j^{(y)}$. Thus the objective function $\mathcal{F}$ is effectively parameterized in terms of the cluster prototypes, and the problem of finding independent clusters now reduces to finding the cluster prototypes that optimize the objective function. The gradient of this function with respect to the prototypes $\mathbf{m}_i^{(x)}$ is given by

$$\nabla_{\mathbf{m}_i^{(x)}}\mathcal{F} = \frac{1}{\ln(2)}\sum_{i'=1}^{r}\sum_{k=1}^{N}\left(\frac{\lambda}{r}\left\{\sum_{j=1}^{c}\left[1+\ln\left(\frac{\sum_{k'=1}^{N} v_{i'}^{(\mathbf{x}_{k'})} v_j^{(\mathbf{y}_{k'})}}{\sum_{k'=1}^{N} v_{i'}^{(\mathbf{x}_{k'})}}\Big/\frac{1}{c}\right)\right]\cdot\left[\frac{v_j^{(\mathbf{y}_k)}}{\sum_{k'=1}^{N} v_{i'}^{(\mathbf{x}_{k'})}} - \frac{\sum_{k'=1}^{N} v_{i'}^{(\mathbf{x}_{k'})} v_j^{(\mathbf{y}_{k'})}}{\sum_{k'=1}^{N}(v_{i'}^{(\mathbf{x}_{k'})})^2}\right]\right\} - \frac{\lambda}{c}\left\{\sum_{j=1}^{c}\left[1+\ln\left(\frac{\sum_{k'=1}^{N} v_{i'}^{(\mathbf{x}_{k'})} v_j^{(\mathbf{y}_{k'})}}{\sum_{k'=1}^{N} v_j^{(\mathbf{y}_{k'})}}\Big/\frac{1}{r}\right)\right]\left[\frac{v_j^{(\mathbf{y}_k)}}{\sum_{k'=1}^{N} v_j^{(\mathbf{y}_{k'})}}\right]\right\} - \frac{1}{N}\left[1+\ln\left(v_{i'}^{(\mathbf{x}_k)}\Big/\frac{1}{r}\right)\right]\right)\nabla_{\mathbf{m}_i^{(x)}} v_{i'}^{(\mathbf{x}_k)}, \quad (13)$$

where

$$\nabla_{\mathbf{m}_i^{(x)}} v_i^{(\mathbf{x}_k)} = \frac{2\rho(\mathbf{x}_k - \mathbf{m}_i^{(x)})}{D} v_i^{(\mathbf{x}_k)}(\delta_{i',i} + v_i^{(\mathbf{x}_k)}). \quad (14)$$

Here $\delta_{i',i}$ is the Kronecker delta. The index variables i, i', and i'' are over the clusters in the $\mathbf{x}$ vectors, j over the clusters in the $\mathbf{y}$ vectors, and k and k' over the data vectors. The gradients with respect to the prototypes $\mathbf{m}_j^{(y)}$ are calculated analogously.

Optimization of $\mathcal{F}$ is performed using the augmented Lagrangian algorithm with simple bound constraints on the cluster prototypes using the FORTRAN package LANCELOT.[20] The initial cluster prototypes are set using individual k-means clusters in each window. The augmented Lagrangian algorithm iteratively improves these initial prototypes till a local minimum of the objective function is attained.

5. Segmentation Algorithm

The approach we take is as follows. A segmentation is a sequence of windows, where each window is maximally dissimilar from its neighboring windows. But what should the windows be? i.e. where should they start and what should their length be? We first create a "pool" of windows and, for every adjacent pair of windows, apply our optimization problem to determine the quality of independent/dissimilar clustering that can result with the given pair of windows. Subsequently, we aim to tile the entire time course by picking windows where adjacent pairs have been whetted by the optimization algorithm to have dissimilar clusterings.

In more detail, let $\mathcal{T} = (t_1, t_2, \ldots, t_l)$ be the given time series data sequence, and $l_{\min}$ and $l_{\max}$ be the minimum and maximum window lengths, respectively. For each time point t_a, we define the set of windows starting from t_a as $S_{t_a} = \{w_{t_a}^{t_b} | l_{\min} \leq t_b - t_a + 1 \leq l_{\max}\}$. Given a window $w_{t_a}^{t_b}$, the choices for the next window are given by $S_{t_{b+1}}$, the set of windows starting form t_{b+1}. These windows can be organized as nodes of a directed acyclic graph, where directed edges exist between $w_{t_a}^{t_b} \in S_{t_a}$ and every $w_{t_{b+1}}^{t_c} \in S_{t_{b+1}}$. The edge weights are set to be the objective function from Eq. (12) realized by simultaneously clustering the windows $w_{t_a}^{t_b}$ and $w_{t_{b+1}}^{t_c}$, as discussed in the previous section. Since local optimization procedures are sensitive to initialization, we perform 100 random restarts of the optimization procedure (each time with different k-means prototypes found in individual windows) and choose the best (minimum) of the local optimum solutions as the weight for the edge between the two windows. Given this weighted directed acyclic graph, the problem of segmenting the time series is equivalent to finding the minimum path (if one exists) between a node representing a window beginning at t_1 and a node corresponding to a window that ends in t_l (recall that there can be several choices for nodes beginning at t_1 as well as for those ending at t_l, depending on $l_{\min}$ and $l_{\max}$). We find the shortest path using dynamic programming (Dijkstra's algorithm) where the path length is defined as D_{avg}, given by Eq. (16), described later.

6. Experiments

6.1. *Datasets*

We analyzed the following datasets using our segmentation algorithm.

YMC: As stated earlier, the YMC dataset[2] consists of 36 time points collected over three continuous cycles.

Table 1. Datasets analyzed and parameters used.

		No. of genes		Parameters		
Dataset	No. time pts.	Original	After filtering	λ	$l_{\min}$	$l_{\max}$
YMC	36	6555	3602	1.4	4	7
YCC	18	6076	2196	1.25	3	5
HP	14	6076	2471	1.55	3	7

YCC: This dataset is taken from experiments performed by Spellman *et al.*[1] We analyzed the data containing the gene expression measurements over two continuous cell cycles after the Yeast cells are released from an α-factor arrest.

HP: This dataset was taken from the experiments conducted by Shapira *et al.*[21] The effects of oxidative stress induced by hydrogen peroxide (HP) on the Yeast cell cycle are studied in these experiments. Adding HP to Yeast cells at 25 minutes after release from G1 arrest leads to a cell cycle arrest in the subsequent G2/M phase.

For all the datasets, we filtered the genes containing missing values and also the genes that do not have an annotation in any GO biological process category (revision 4.205 of GO released on 14 March 2007). We then used the parameters as shown in Table 1 to segment the datasets. For all the datasets, we ranged the number of clusters in each window between 3 and 15, and the λ value was adjusted to give approximately equal sized clusters with good intra-cluster similarities.

6.2. *Evaluation metrics*

We evaluate our clusterings and segmentations in five ways: cluster stability, cluster reproducibility, functional enrichment, segmentation quality, and segmentation sensitivity. Cluster stability and cluster reproducibility are used to filter the patterns obtained by the segmentation algorithm whereas the other three criteria are used to evaluate the quality of segmentation.

We assess **cluster stability** using a bootstrap procedure to determine the significance of genes brought together. Recall that each window except the first and last windows has two sets of clusters, one set independent with respect to the previous window and the other independent with respect to the next window. We are interested in the genes that are significantly clustered together in these two sets of clusters, as they represent the genes that are specific to the window under consideration. We calculate a contingency table between these two clusterings for each window (excluding the first and the last windows) in which each cell represents the number of genes that are together across the two independent sets of clusters. We randomly sample 1000 pairs of clusterings within each window (with cluster sizes the same as the two independent clusterings) and compute their contingency tables. By the central limit theorem, the distribution of counts in each cell of the table is approximately normal (also verified using a Shapiro-Wilk normality test with $p = 0.05$). We now evaluate each cell of the actual contingency table with

respect to the corresponding random distribution and retain only those cells that have more genes than that observed at random with $p < 0.05$ (Bonferroni corrected with the number of cross clusters to account for multiple hypothesis testing). To ensure **reproducibility** of clusters, we retain only those genes in each significant cell of the contingency table that are together in more than 150 of the 200 clusterings (conducted with different initializations). For the first and last windows, which have only 100 randomly initialized clusterings, we retain those genes that are clustered together in more than 75 of the 100 clusterings. At this stage, for each segment we obtain a contingency table with significant cells representing the group of genes specific to the particular segment, which we can refer to as cross-cluster. We perform **functional enrichment** using the GO biological process ontology (since we are tracking biological processes) over each of these cross clusters. A hypergeometric p-value is calculated for each GO biological process term, and an appropriate cutoff is chosen using a false discovery rate (FDR) q-level of 0.01.

The **segmentation quality** is calculated as a partition distance[22] between the "true" segmentation (from the literature of the YMC and YCC) to the segmentations computed by our algorithm. We view each window as a set of time points so that a segmentation is a partition of time points. Given two segmentations S_1 and S_2, whose windows are indexed by the variables $w_{t_a}^{t_b}$ and $z_{t_c}^{t_d}$ respectively, the partition distance is given by:

$$P_D = -\sum_{w_{t_a}^{t_b} \in S_1} \sum_{z_{t_c}^{t_d} \in S_2} |w_{t_a}^{t_b} \cap z_{t_c}^{t_d}| \ \log_2 \frac{|w_{t_a}^{t_b} \cap z_{t_c}^{t_d}|}{|w_{t_a}^{t_b}|} - \sum_{z_{t_c}^{t_d} \in S_2} \sum_{w_{t_a}^{t_b} \in S_1} |w_{t_a}^{t_b} \cap z_{t_c}^{t_d}| \ \log_2 \frac{|w_{t_a}^{t_b} \cap z_{t_c}^{t_d}|}{|z_{t_c}^{t_d}|}. \tag{15}$$

The **segmentation sensitivity** to variations in the number of clusters is calculated as the average of the ratios of KL-divergences between the segments to the maximum possible KL divergence between those segments. This latter figure is easy to compute as a function of the number of clusters, which is considered uniform throughout the segmentation. Suppose we have $|S|$ windows in a given segmentation $S = \{w_{t_1}^{t_a}, w_{t_{a+1}}^{t_b}, \ldots, w_{t_{j+1}}^{t_k}, w_{t_{k+1}}^{t_l}\}$ with c clusters in each window. Let $\mathcal{F}_{\max}$ be the objective function value for the maximally similar clustering (the $c \times c$ diagonal contingency table (b) in the example in Sec. 3). Then the measure we compute is

$$D_{\text{avg}} = \frac{1}{|S| - 1} \left[\frac{\mathcal{F}_{\{w_{t_1}^{t_a}, w_{t_{a+1}}^{t_b}\}}}{\mathcal{F}_{\max}} + \frac{\mathcal{F}_{\{w_{t_{b+1}}^{t_c}, w_{t_{c+1}}^{t_d}\}}}{\mathcal{F}_{\max}} + \cdots + \frac{\mathcal{F}_{\{w_{t_j}^{t_k}, w_{t_{k+1}}^{t_l}\}}}{\mathcal{F}_{\max}} \right], \tag{16}$$

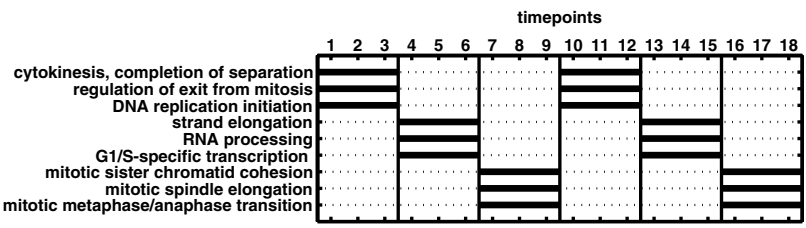
where $\mathcal{F}_{\{w_{t_a}^{t_b}, w_{t_{b+1}}^{t_c}\}}$ is the optimal objective function value obtained by clustering the pair of adjacent windows $w_{t_a}^{t_b}$, $w_{t_{b+1}}^{t_c}$. Observe that this criterion is different from the actual criterion optimized during the segmentation. D_{avg} compares our segmentation (which identifies dissimilar sets of clusters) to the case when there are exactly similar clusters. Lower values of this ratio indicate that the segmentation

captures maximal independence between adjacent segments while higher values indicate that the clusters obtained are more similar in adjacent segments.

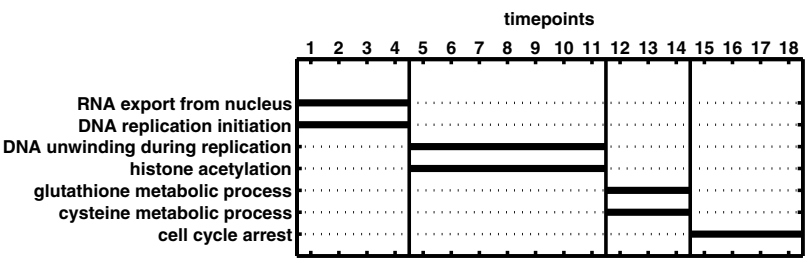
6.3. *Results*

YMC: The segmentation generated for the minimum number (3) of clusters is: 1–6, 7–10, 11–14, 15–18, 19–22, 23–26, 27–31, 32–36, which correspond to alternating R/B, R/C, and Ox phases. The GO categories enriched ($p < 10^{-7}$) are depicted in Fig. 1 (bottom).

YCC: The segmentation [Fig. 2(a)] generated for YCC — 1–3, 4–6, 7–9, 10–12, 13–15, 16–18 — is also periodic with the stages approximately corresponding to alternating M/G1, {G1,S}, {G2,M} phases. Note that each phase is of very short length in this experiment as compared to YMC: the phases M/G1, G1, S each last for approximately two time points, while the G2 phase lasts only for one time point. Because our minimum window length is three (set so that we recover significant clusterings and regroupings), we cannot resolve these short-lived phases. A possible approach is to use continuous representations such as spline fits to gain greater resolution of data sampling. Nevertheless, the key events occurring in these segments are retrieved with high specificity ($p < 10^{-7}$) as shown in Fig. 2(a).

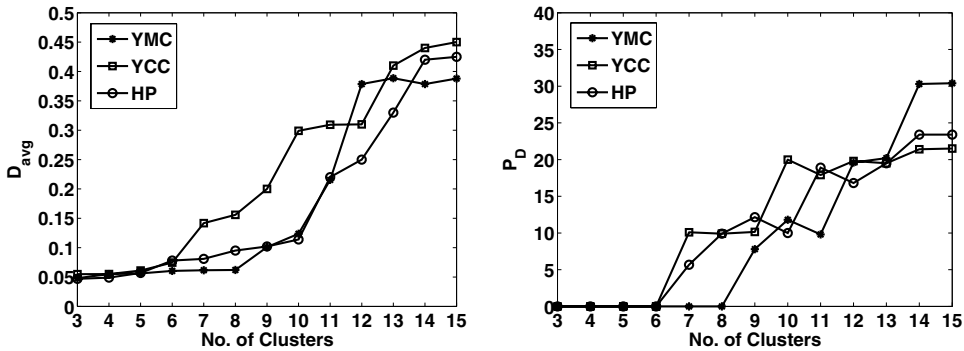


(a)



(b)

Fig. 2. Gantt charts depicting the segments and enriched GO categories: (a) YCC, (b) HP. The bars in the Gantt chart represent functionally enriched GO categories in the cross-clusters of the segements. Only a few GO categories are depicted due to lack of space.



(a) (b)

Fig. 3. (a) Average ratio of the objective function values for the contingency tables to the maximum possible value. (b) Distance of the segmentation recovered to the "true" segmentation.

HP: The segmentation obtained is 1–4, 5–11, 12–14, 15–20, corresponding to G1, S, G2, G2/M phases of the cell cycle as depicted in Fig. 2(b). Note that the cells here are arrested in the G2/M phase.

Effect of number of clusters: The sensitivitily of segmentation output to change in the number of clusters is studied in Fig. 3. In Fig. 3(a), we see that as the number of clusters increases, it is increasingly difficult to obtain independent clusterings, and hence for higher values of the number of clusters, the segmentation problem actually resembles associative clustering (observe that this curve tends toward a D_{avg} value of 0.5). Figure 3(b) tracks the segmentation quality, and shows that the correct segmentation is recovered for many settings in the lower range for number of clusters, but as the number of clusters increases, the best segmentations considerably deviate from the true segmentation. Nevertheless, comparing the two plots, we see that D_{avg} tracks the segmentation quality P_D well and hence can be a useful surrogate for determining the "right" number of clusters.

Biological significance of results: One of the applications of our methods is to decode temporal relationships between biological processes. Since cell division processes are enriched in both YCC and YMC, we superimposed those segments of our two Gantt charts [from Fig. 1 and Fig. 2(a)], and observed that the oxidative metabolism phase of YMC typically precedes the transition from G1 to S in the YCC. This is significant because it permits the DNA replication process to occur in a reductive environment. These and other connections between the YMC and the YCC are presently under intense investigation,[23–25] and hypotheses involving biochemical process compatibility versus coordinated metabolic "bursts" are currently being compared and contrasted. These methodologies can be used to evaluate transcriptional profiles for direct comparison with proteomic and metabolic profiling datasets,[25] permitting a systems-biology perspective of yeast cellular dynamics.

Such work can also lead to the design of targeted biological experiments aimed at determining the central players in transition from one individual state to another, as well as the evaluation of similar temporal shifts in other organisms. Also note that in the case of HP dataset, the biological processes include glutathione and cysteine metabolic processes which eventually lead to cell cycle arrest as indicated by Shapira *et al.*[21]

7. Discussion

We have presented a novel approach to simultaneously segment multiple time course data using clusterset dissimilarity as a driving criterion for optimization. Temporal modeling of biological process activity is a burgeoning area of research. In Ref. 26, for instance, the authors use a HMM for modeling transitions between biological processes (which are assumed to be the hidden, unobserved, variables) and also estimate gene-process association matrices along with activity levels of different processes. However, since the processes are not *a priori* defined, this approach uses multiple time course datasets to estimate gene-process association matrices with high fidelity and, unlike our approach, does not result in a segmentation of a given dataset. Nevertheless, the ideas of activity-level modeling from Ref. 26 and our emphasis on segmentation through cluster dynamics can be fruitfully combined in a single framework. In particular, we can develop richer models of cluster reorganization, e.g. dynamic revisions in the number of clusters, split-and-merge behavior of clusters, and a HMM for cluster reorganization, leading to inference of complete temporal logic models.

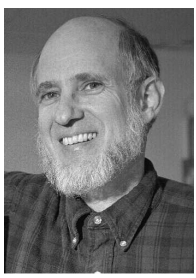
References

1. Spellman PT, Iyer VR, Anders K, Eisen MB, Brown PO, Botstein D, Futcher B, Comprehensive Identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization, *Mol Biol Cell* **9**:3273–3297, 1998.
2. Tu BP, Kudlicki A, Rowicka M, McKnight SL, Logic of the yeast metabolic cycle: Temporal compartmentalization of cellular processes, *Science* **310**:1152–1158, 2005.
3. Lund J, Tedesco P, Duke K, Wang J, Kim SK, Johnson TE, Transcriptional profile of aging in *C. elegans*, *Curr Biol* **12**:1566–1573.
4. Bar-Joseph Z, Gerber G, Gifford DK, Jaakkola T, Simon I, Continuous representations of time-series gene expression data, *J Comput Biol* **10**:341–356, 2003.
5. Schliep A, Schonhuth A, Steinhoff C, Using hidden Markov models to analyze gene expression time course data, *Bioinformatics* **19**:i255–i263, 2003.
6. Simon I, Siegfried Z, Ernst J, Bar-Joseph Z, Combined static and dynamic analysis for determining the quality of time-series expression profiles, *Nat Biotech* **23**:1503–1508, 2005.
7. Singh R, Palmer N, Gifford D, Berger B, Bar-Joseph Z, Active learning for sampling in time-series experiments with application to gene expression analysis, *in Proc Int Conf Machine Learning*, 832–839, 2005.
8. Ernst J, Nau GJ, Bar-Joseph Z, Clustering short time series gene expression data, *Bioinformatics* **21**:i159–i168, 2005.

9. Yoneya T, Mamitsuka H, A hidden Markov model-based approach for identifying timing differences in gene expression under different experimental factors, *Bioinformatics* **23**:842–849, 2007.
10. Shi Y, Mitchell T, Bar-Joseph Z, Inferring pairwise regulatory relationships from multiple time series datasets, *Bioinformatics* **23**:755–763, 2007.
11. Keogh E, Chu S, Hart D, Pazzani M, Segmenting time series: A survey and novel approach, *Data Mining in Time Series Databases*, World Scientific, Singapore, 2003.
12. Fearnhead P, Exact and efficient Bayesian inference for multiple changepoint problems, *Stat Comput* **16**:203–213, 2006.
13. Abonyi J, Feil B, Nemeth S, Arva P, Fuzzy clustering based segmentation of time-series, in *Advances in Intelligent Data Analysis V*, pp. 275–285, Springer, Berlin/Heidelberg, 2003.
14. Xuan X, Murphy K, Modeling changing dependency structure in multivariate time series, *in Proc Int Conf Machine Learning*, 1055–1062, 2007.
15. Mörchen F, Ultsch A, Discovering temporal knowledge in multivariate time series, *Proc GfKI*, pp. 272–270, Berlin, 2005.
16. Kullback S, Gokhale DV, *The Information in Contingency Tables*, Marcel Dekker Inc., 1978.
17. Kreisselmeier G, Steinhauser R, Systematic control design by optimizing a vector performance index, *Proc IFAC Symp Computer Aided Design Control Systems*, 113–117, 1979.
18. Barthelemy JFM, Riley MF, Improved multi-level optimization approach for the design of complex engineering systems, *AIAA J* **26**:353–360, 1988.
19. Banerjee A, Ghosh J, Scalable clustering algorithms with balancing constraints, *Data Min Knowl Discov* **13**:365–395, 2006.
20. Conn AR, Gould NIM, Toint PL, *LANCELOT: A Fortran Package for Large-scale Nonlinear Optimization (Release A)*, Springer, 1992.
21. Shapira M, Segal E, Botstein D, Disruption of yeast forkhead-associated cell cycle transcription by oxidative stress, *Mol Biol of the Cell* **15**:5659–5669, 2004.
22. De Mántaras RL, A distance-based attribute selection measure for decision tree induction, *Machine Learning* **6**:81–92, 1991.
23. Chen Z, Odstrcil EA, Tu BP, McKnight SL, Restriction of DNA replication to the reductive phase of the metabolic cycle protects genome integrity, *Science* **316**:1916–1919, 2007.
24. Futcher B, Metabolic cycle, cell cycle, and the finishing kick to start, *Genome Biol* **7**:107–111, 2006.
25. Murray DB, Beckmann M, Kitano H, Regulation of yeast oscillatory dynamics, *Proc Natl Acad Sci USA* **104**:2241–2246, 2007.
26. Shi Y, Klustein M, Simon I, Mitchell T, Bar-Joseph Z, Continuous hidden process model for time series expression experiments, *Bioinformatics* **23**:i459–i467, 2007.

Satish Tadepalli is a Ph.D. student in the Department of Computer Science at Virginia Tech., USA. He received his M.S. in Computer Science from Virginia Tech in 2003. His research interests include data mining, computational biology, and graphical models. He has worked at Novartis Pharamceuticals, and Rogers Casey.

Naren Ramakrishnan is a Professor and Director of Graduate Studies in the Department of Computer Science at Virginia Tech., USA. His research interests include problem solving environments, mining scientific data, and information personalized. He received a Ph.D. in Computer Sciences from Purdue University.



Layne T. Watson received a Ph.D. in Mathematics from the University of Michigan, Ann Arbor, in 1974. His research interests include numerical analysis, optimization, parallel computation, and bioinformatics. He has worked at Sandia National Laboratories, University of Michigan, and Michigan State University.

Bud Mishra is a Professor of Computer Science and Mathematics at NYU, Professor of Human Genetics at Mt. Sinai School of Medicine, and a Professor of Cell Biology at NYU School of Medicine. He has a Ph.D. in Computer Science from Carnegie-Mellon University and serves as the editor of several journals.

Richard F. Helm is an Associate Professor of Biochemistry at Virginia Tech. His research focus is on understanding the processes used by organisms to turn off metabolic activity. His areas of expertise include analytical biochemistry, as well as carbohydrate chemistry. He is the Director of the Virginia Tech mass spectrometry incubator.