Simultaneously Segmenting Multiple Gene Expression Time Courses by Analyzing Cluster Dynamics

Satish Tadepalli[1,*], Naren Ramakrishnan [1], Layne T. Watson[1,2],
Bhubaneshwar Mishra[3], and Richard F. Helm[4]

[1]*Department of Computer Science,* [2]*Department of Mathematics,* [4]*Department of Biochemistry*
Virginia Polytechnic Institute and State University , Blacksburg, VA 24061
[*]*E-mail: stadepal@vt.edu*
www.vt.edu
[3]*Courant Institute of Mathematical Sciences, New York University, NY 10012*
www.nyu.edu

We present a new approach to segmenting multiple time series by analyzing the dynamics of cluster rearrangement around putative segment boundaries. 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. This approach finds application in distilling large numbers of gene expression profiles into temporal relationships underlying biological processes. 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.

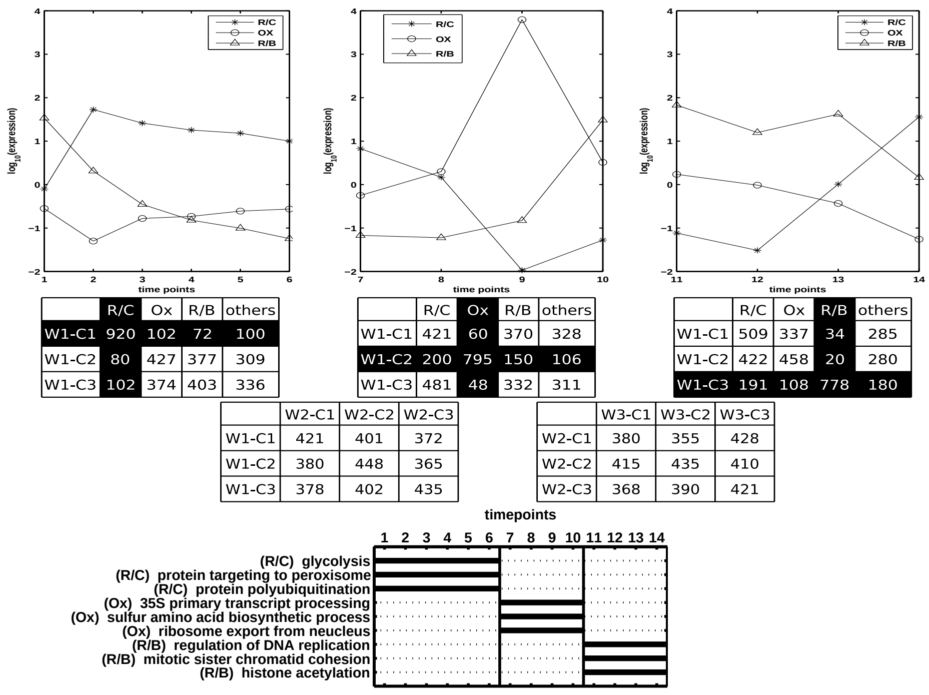
Keywords: Time series segmentation, clustering, KL-divergence, temporal regulation.

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. Recent research efforts have considered using static measurements to "fill in the gaps" in the time series data,[3] quantifying timing differences in gene expression,[4] and reconstructing regulatory relationships.[5]

One of the attractions of time series analysis is its promise to reveal temporal relationships underlying biological processes:[6] which process occurs before which other, and what are the "checkpoints" that must be satisfied (and when). Although similar analysis can also 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 yet characterized *a priori.* In particular, given multiple gene expression profiles over a time course, we desire to identify both segments of the time course where groups show concerted behavior and boundaries between segments where there is significant functional "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.

It is important to contrast our goals with prior work. Typical published results on time series segmentation[7] are focused on segmenting a single time series with homogeneity assumptions on successive time points. We are focused on simultaneously segmenting multiple time series by modeling 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 reorganizations underlying the dataset. As an example, consider the yeast metabolic cycle (YMC), using the dataset of



	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
W1-C1	421	60	370	328
W1-C2	200	795	150	106
W1-C3	481	48	332	311

	R/C	Ox	R/B	others
W1-C1	509	337	34	285
W1-C2	422	458	20	280
W1-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 the segments in the Yeast metabolic cycle.

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.* 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 hour period using our segmentation algorithm and arrived at the same segmentation corresponding to three cycles, one of which is shown in Fig. 1. The top row depicts the prototypes of the clusters in the **R/C**, **Ox**, and **R/B** phases respectively. The second row shows how the genes corresponding to each phase are coming together in a particular cluster *within a segment* (intersection of the highlighted row and column) while the genes of other phases are spread apart across other clusters. This regrouping of the clusters of genes *across the segments* is captured by the contingency tables in the third row. Observe that the contingency tables in the second row involve significant enrichments whereas the tables in the third row approximate a uniform distribution. These time-bounded enrichments for the clusters in each segment identify the biological processes with modulated activities as shown in the Gantt chart in the bottom row of Fig. 1. 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.

2. Problem Formulation

We are given multiple l-vectors of measurements $\mathcal{G} = \{\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_\nu\}$, 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 sequence 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 $w_{t_{b+1}}^{t_c}$. For instance, given a dataset with 18 genes and 3 clusters in either window, the evaluation criterion prefers contingency table (a) below over tables (b) and (c).

$$\boxed{\begin{matrix}2&2&2\\2&2&2\\2&2&2\end{matrix}} \qquad \boxed{\begin{matrix}6&0&0\\0&6&0\\0&0&6\end{matrix}} \qquad \boxed{\begin{matrix}0&6&0\\6&0&0\\0&0&6\end{matrix}}$$

(a) (b) (c)

Here the rows refer to clusters of $w_{t_a}^{t_b}$ and the columns refer to clusters of $w_{t_{b+1}}^{t_c}$. We achieve this by enforcing that the (projected) row-wise and column-wise distributions from the contingency table resemble a uniform distribution. Formally, given two windows $w_{t_a}^{t_b}$ and $w_{t_{b+1}}^{t_c}$, which have been clustered into r and c clusters (respectively), we define the $r \times c$ contingency table over the clusterings. Entry n_{ij} in the $(i,j)^{th}$ cell of the table represents the overlap between the genes clustered together in cluster i of $w_{t_a}^{t_b}$ and in cluster j of $w_{t_{b+1}}^{t_c}$. The sizes of the clusters in $w_{t_a}^{t_b}$

are given by the column-wise sums across each row: $n_{i\cdot} = \sum_j n_{ij}$, while the sizes of clusters in $w_{t_{b+1}}^{t_c}$ are given by row-wise sums down each column: $n_{\cdot j} = \sum_i n_{ij}$. Using these, we define $(r + c)$ probability distributions, one for each row and one for each column. The distribution corresponding to row i, $\mathbf{R}_i$, takes values from the column indices, i.e., $1, \ldots, c$, with value j $(1 \leq j \leq c)$ occurring with probability $\frac{n_{ij}}{n_{i\cdot}}$. Similarly, the column distribution for column j, $\mathbf{C}_j$, takes values from the row indices, i.e., $1, \ldots, r$, with value i $(1 \leq i \leq r)$ occurring with probability $\frac{n_{ij}}{n_{\cdot j}}$. We capture the deviation of these row-wise and column-wise distributions w.r.t. the uniform distribution as

$$\mathcal{F} = \frac{1}{r}\sum_{i=1}^{r} D_{KL}(\mathbf{R}_i || U(\frac{1}{c})) + \frac{1}{c}\sum_{j=1}^{c} D_{KL}(\mathbf{C}_j || U(\frac{1}{r})), \tag{1}$$

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 $p(x)$ and $q(x)$, and $U(\cdot)$ denotes the uniform distribution whose argument is the probability of any outcome. The optimization problem is then to minimize $\mathcal{F}$. This function can also be interpreted in terms of entropies of the row-wise and column-wise distributions, and also in terms of conditional entropies of the clusters in windows $w_{t_a}^{t_b}$ and $w_{t_{b+1}}^{t_c}$. Also, $\mathcal{F}$ has connections to the principle of minimum discrimination information (MDI).[8] 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. Observe that the combinations of the r row-wise KL-divergences and c column-wise KL-divergences are averaged to form $\mathcal{F}$. This is to mitigate the effect of lopsided contingency tables ($r \gg c$ or $c \gg r$) wherein 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. (1) can be readily extended to the case where we have more than two segments.

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. Maximizing $\mathcal{F}$ leads to highly dependent clusters across the windows which is the same as associative clustering described by Kaski *et al.*[9] However, for our current problem of time series segmentation, we are concerned with only minimizing $\mathcal{F}$ to obtain independent clusters.

3. 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. In typical clustering algorithms, each cluster has a prototype and the data vectors are assigned to the nearest cluster based on some distance measure from these prototypes. The prototypes are iteratively improved to find the best possible clusters.

Again, 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 a $r \times c$ contingency table. Let $\mathbf{m}_i^{(x)}$ be the prototype vector for the ith cluster of the $\mathbf{x}$ vectors. The assignment of a data vector to the clusters is given by the probability distribution $\mathbf{V}(\mathbf{x}_k) = \{V_i(\mathbf{x}_k)\}$, where $\sum_{i=1}^{r} V_i(\mathbf{x}_k) = 1$. The probabilities $V_i(\mathbf{x}_k)$ are the cluster membership indicator variables, i.e., the probability that data vector k is assigned to cluster i. Similar cluster prototypes $\mathbf{m}_j^{(y)}$, distributions $\mathbf{V}(\mathbf{y}_k)$, and cluster indicator variables $V_j(\mathbf{y}_k)$ are defined for $\mathbf{y}$ vectors as well. Then the contingency table counts can be calculated as $n_{ij} = \sum_k V_i(\mathbf{x}_k)V_j(\mathbf{y}_k)$. Assigning a data vector to the nearest cluster with a probability of one and calculating n_{ij} renders the objective function $\mathcal{F}$ in Eq. (1) 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 parametrized as a continuously differentiable function that assigns each data vector to its nearest cluster with a probability close to one and to the other clusters with a probability close to zero, i.e. $V_i(\mathbf{x}_k), V_j(\mathbf{y}_k) \in (0, 1)$. For this purpose, we define

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

where, $D = \max\limits_{k,k'} ||\mathbf{x}_k - \mathbf{x}_{k'}||^2, 1 \le k, k' \le \nu$ is the pointset diameter. A well known approximation to $\min\limits_{i'} \gamma_{(i,i')}(\mathbf{x}_k)$ is the Kreisselmeier-Steinhauser (KS) envelope function[10] given by

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

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

$$V_i(\mathbf{x}_k) = Z(\mathbf{x})^{-1} \exp \Big[\rho\, KS_i(\mathbf{x}_k) \Big],$$

where $Z(\mathbf{x})$ is a normalizing function such that $\sum_{i=1}^{r} V_i(\mathbf{x}_k) = 1$. The cluster membership indicators for the "right" window, $V_j(\mathbf{y}_k)$, are also smoothed similarly. The membership probability of a vector to a cluster is assigned relative to its distance from all the other clusters as captured by the function γ in Eq. (2). This approach tracks the membership probabilities better than using individual Gaussians for each cluster as suggested by Kaski et al[9].

Minimizing the function $\mathcal{F}$ in Eq. (1) 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 vector is assigned with uniform probability to every cluster. Recall the contingency table example from Sec. 2; each of the 18 genes 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 each data vector's cluster membership 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

$$\begin{aligned}\mathcal{F} =& \frac{\lambda}{r}\sum_{i=1}^{r} D_{KL}(\mathbf{R}_i||U(\frac{1}{c})) + \frac{\lambda}{c}\sum_{j=1}^{c} D_{KL}(\mathbf{C}_j||U(\frac{1}{r})) \\ &- \frac{1}{\nu}\sum_{k=1}^{\nu} D_{KL}(\mathbf{V}(\mathbf{x}_k)||U(\frac{1}{r})) - \frac{1}{\nu}\sum_{k=1}^{\nu} D_{KL}(\mathbf{V}(\mathbf{y}_k)||U(\frac{1}{c})),\end{aligned} \tag{3}$$

where λ is the weight, set to a value greater than 1, gives more emphasis to minimizing the row and column wise distributions. This also enforces equal cluster sizes. Optimization of $\mathcal{F}$ is performed using the augmented Lagrangian algorithm with simple bound constraints on the cluster prototypes using the FORTRAN package LANCELOT.[11] The initial cluster prototypes are set using individual k-means clusters in each window and are iteratively improved till a local minimum of the objective function is attained.

4. Segmentation Algorithm

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 $S_{t_a} = \{w_{t_a}^{t_b} | l_{min} \le t_b - t_a + 1 \le l_{max}\}$. 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. (3) 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. (4), described later.

5. Experiments

Datasets: Our experimental datasets constitute gene expression measurements spanning the yeast metabolic cycle (YMC) and the yeast cell cycle (YCC). As stated earlier, the YMC dataset[2] consists of 36 time points collected over three continuous cycles. The YCC was taken from the well known α-factor experiment of Spellman et al.[1] The original YMC dataset consists of 6555 unique genes from the *S. cerevisiae* genome. We first eliminated those genes that do not have an annotation in any GO biological process category (revision 4.205 of GO released on 14 March 2007), resulting in a universal set of 3602 genes. The gene expression values were log transformed (base 10) and normalized such that the mean expression of each gene across all time points is zero. To segment this dataset we experimented with the number of clusters in each segment ranging from three to 15, $l_{min} = 4,\ l_{max} = 7,\ \rho = 20$, and $\lambda = 1.4$. The λ and ρ values were adjusted to give approximately equal sized clusters with good intracluster similarities. For the YCC dataset which originally had 6076 genes, we considered the genes with no missing values and mean centered each gene's expression across all time points to zero. From this data, we removed the genes that do not have any annotation in the GO biological process category resulting in a final set of 2196 genes. To segment this dataset, again we ranged from three to 15 clusters in each window, $l_{min} = 3,\ l_{max} = 5,\ \rho = 20$, and $\lambda = 1.4$ (ρ and λ adjusted as before).
Evaluation metrics: We evaluate our clusterings and segmentations in five ways: cluster stability, cluster reproducibility, functional enrichment, segmentation quality, and segmentation sensitivity. We assess **cluster stability** using a bootstrap procedure to determine 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 window). Each cell in the contingency table 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 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. After the above two steps, we perform **functional enrichment** using the GO biological process ontology (since we are tracking biological processes) over the selected clusters of genes. 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[12] 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:

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

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. 2). Then the measure we compute is

$$D_{avg} = \frac{1}{|S|-1} \left[\frac{\mathcal{F}_{\{w_{t_1}^{t_a}, w_{t_{a+1}}^{t_b}\}}}{\mathcal{F}_{max}} + \frac{\mathcal{F}_{\{w_{t_{b+1}}^{t_c}, w_{t_{c+1}}^{t_d}\}}}{\mathcal{F}_{max}} + \ldots + \frac{\mathcal{F}_{\{w_{t_j}^{t_k}, w_{t_{k+1}}^{t_l}\}}}{\mathcal{F}_{max}} \right], \quad (4)$$

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 $w_{t_a}^{t_b}, w_{t_{b+1}}^{t_c}$. Lower values of this ratio indicate that the segmentation captures maximal independence between adjacent segments while higher values indicate the clusters obtained are more similar in adjacent segments.

Results: The YMC 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/C, Ox, and R/B phases. The GO categories enriched ($p < 10^{-7}$) in one cycle for this dataset have already been depicted in Fig. 1. This segmentation is stable up to eight clusters after which it begins to deviate from the "true" segmentation (discussed further below). The segmentation (Fig. 2) 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

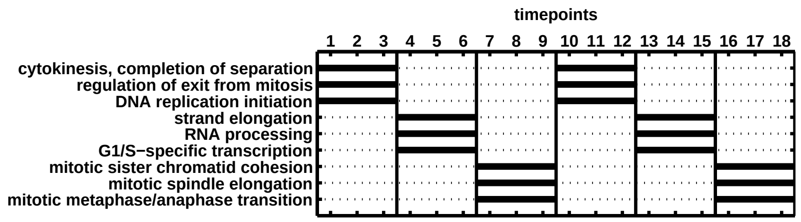


Fig. 2. Gantt chart from segmentation of Spellman et al. dataset. To preserve space, only some of the enriched GO biological process terms are shown.

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.

The effect of the number of clusters on segmentation characteristics is studied

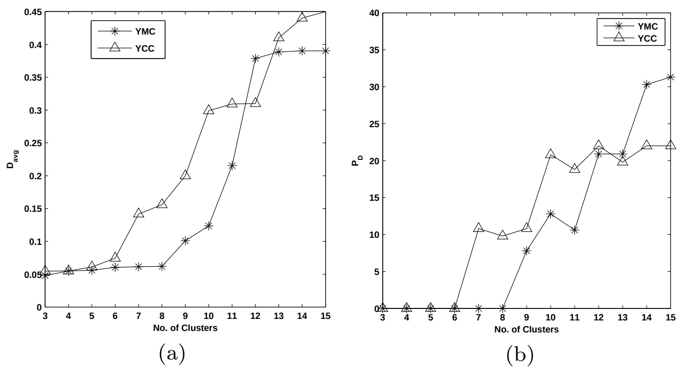


Fig. 3. (a)Segmentation sensitivity and (b)Segmentation quality

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.

6. Discussion

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), 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.[13–15]

Temporal modeling of biological process activity is a burgeoning area of research. For instance, Shi *et al.*[16] present an approach to detect the activity levels of biological processes in a time series dataset. Such ideas can be combined with our segmentation algorithm to get a temporal activity level model of biological processes. In particular, we can develop richer models of cluster reorganization, e.g., dynamic revisions in the number of clusters, split-and-merge behaviors of clusters, and a HMM for cluster re-organization, leading to inference of complete temporal logic models.

References

1. P. Spellman, G. Sherlock, M. Zhang, V. Iyer, K. Anders, M. Eisen, P. Brown, D. Botstein and B. Futcher, *Molecular Biology of the Cell* **9**, 3273 (1998).
2. B. Tu, A. Kudlicki, M. Rowicka and S. McKnight, *Science* **310**, 1152 (2005).
3. I. Simon, Z. Siegfried, J. Ernst and Z. Bar-Joseph, *Nature Biotechnology* **23**, 1503 (2005).
4. T. Yoneya and H. Mamitsuka, *Bioinformatics* **23**, 842 (2007).
5. Y. Shi, T. Mitchell and Z. Bar-Joseph, *Bioinformatics* **23**, 755 (2007).
6. N. Ramakrishnan, M. Antoniotti and B. Mishra, Reconstructing formal temporal logic models of cellular events using the go process ontology, in *Proceedings of the Eighth Annual Bio-Ontologies Meeting (ISMB'05 Satellite Workshop)*,
7. E. Keogh, S. Chu, D. Hart and M. Pazzani, Segmenting time series: A survey and novel approach, in *Data Mining in Time Series Databases*, (World Scientific Publishing Company, 2003)
8. S. Kullback and D. Gokhale, *The Information in Contingency Tables* (Marcel Dekker Inc., 1978).
9. S. Kaski, J. Nikkilä, J. Sinkkonen, L. Lahti, J. Knuuttila and C. Roos, *IEEE/ACM TCBB* **2**, 203 (2005).
10. G. Wrenn, An Indirect Method for Numerical Optimization using the Kreisselmeier-Steinhauser Function NASA Contractor Report 4220(March, 1989).
11. A. Conn, N. Gould and P. Toint, *LANCELOT: A Fortran Package for Large-scale Nonlinear Optimization (Release A)* (Springer Verlag, 1992).
12. R. D. Mántaras, *Machine Learning* **6**, 81 (1991).
13. B. Futcher, *Genome Biology* **7**, 107 (2006).
14. Z. Chen, E. Odstrcil, B. Tu and S. McKnight, *Science* **316**, 1916 (2007).
15. D. Murray, M.Beckmann and H. Kitano, *PNAS* **104**, 2241 (2007).
16. Y. Shi, M. Klustein, I. Simon, T. Mitchell and Z. Bar-Joseph, *Bioinformatics (Proceedings of ISMB 2006)* **23**, i459 (2007).