Analyzing Gene Expression Time-Courses

Alexander Schliep, Ivan G. Costa, Christine Steinhoff, and Alexander Schönhuth

Abstract—Measuring gene expression over time can provide important insights into basic cellular processes. Identifying groups of genes with similar expression time-courses is a crucial first step in the analysis. As biologically relevant groups frequently overlap, due to genes having several distinct roles in those cellular processes, this is a difficult problem for classical clustering methods. We use a mixture model to circumvent this principal problem, with hidden Markov models (HMMs) as effective and flexible components. We show that the ensuing estimation problem can be addressed with additional labeled data—partially supervised learning of mixtures—through a modification of the Expectation-Maximization (EM) algorithm. Good starting points for the mixture estimation are obtained through a modification to Bayesian model merging, which allows us to learn a collection of initial HMMs. We infer groups from mixtures with a simple information-theoretic decoding heuristic, which quantifies the level of ambiguity in group assignment. The effectiveness is shown with high-quality annotation data. As the HMMs we propose capture asynchronous behavior by design, the groups we find are also asynchronous. Synchronous subgroups are obtained from a novel algorithm based on Viterbi paths. We show the suitability of our HMM mixture approach on biological and simulated data and through the favorable comparison with previous approaches. A software implementing the method is freely available under the GPL from http://ghmm.org/gql.

Index Terms—Mixture modeling, hidden Markov models, partially supervised learning, gene expression, time-course analysis.

1 Introduction

Microarrays have changed the way science is done in molecular biology greatly over the course of the last 15 years. In essence, a microarray experiment *quantifies* the individual abundances of mRNA molecules specified a priori in *parallel*. The reaction observed in the experiment is the hybridization of *targets*, the mRNA molecules, to prescribed *probes*, short oligonucleotides or cDNAs. If the targets are the transcripts of genes, the outcome of an experiment can measure genome-wide levels of gene expression.

As a consequence of the high throughput of microarrays, the levels of error and noise in the measurements are high and methods used for the analysis have to reflect that. Normalization of the data is a usual preprocessing step. It is is still subject to debate and an overview of the issues can be found in [1]. More importantly, the biology governing the levels of gene expression we observe is highly complex and still under investigation. Contributing factors and relevant processes exist on many levels of the biological system under investigation *and* its environment. Genes can be involved in several pathways and have multiple functions depending on specifics of the cell's environment. Hence, groups of genes defined according to similarity of function or regulation of a gene are *not* disjoint in general.

We are concerned with microarray experiments which measure levels of gene expression over time. This includes, for example, changes in expression during the cell-cycle, during development or in response to external factors. Finding groups of genes going through the same transcriptional program—having the same pattern of changes in gene expression over time—is a routine approach for a first analysis of such data. Often, clustering, or unsupervised learning, methods are applied to normalized data to identify groups of genes, to organize the data for further manual analysis, and obtain hints about possible function or regulation for groups containing well-studied genes.

1.1 Prior Work on Analyzing Expression Time-Courses

Previous approaches, see [2] for a recent overview, fall roughly into two classes, depending on whether they assume the experiments at different time-points to be independent or not. Methods in the first class neglect these so called *temporal dependencies*. They define a distance measure and group genes in a way that minimizes an objective function based on the distances between expression time-courses. Examples are hierarchical [3], [4] and k-means clustering [5] or singular value decomposition [6]. Note that, for those methods, permuting time points arbitrarily does not change the result of the clustering.

Most methods in the second class are model-based. Statistical models are used to represent clusters, and cluster membership is decided based on maximizing the likelihood of data points. Model-based clustering is more suitable for time-series data [7] and its main advantage is that no distance function between time-courses is required. Euclidean distance, for example, overly emphasizes many noncritical variances of signals—a delay or a slower rate—and similar arguments can be made against other distance functions. Methods used for analysis of expression time-courses are based on cubic splines [8], autoregressive curves [9], [10] or on multivariate Gaussians [11].

Another important criterion is whether the method assumes cyclic time-courses. Methods which do so [12],

• *A. Schliep, I.G. Costa, and C. Steinhoff are with the Max Planck Institute for Molecular Genetics, Ihnestrasse 73, 14195 Berlin, Germany.*
E-mail: {alexander.schliep, ivan.filho, christine.steinhoff}@molgen.mpg.de.
• *A. Schönhuth is with the Center for Applied Computer Science, University of Cologne, Weyertal 80, 50931 Cologne, Germany.*
E-mail: schoenhuth@zpr.uni-koeln.de.

Manuscript received 6 Sept. 2004; revised 3 Dec. 2004; accepted 11 Mar. 2005; published online 31 Aug. 2005.
For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBBSI-0135-0904.

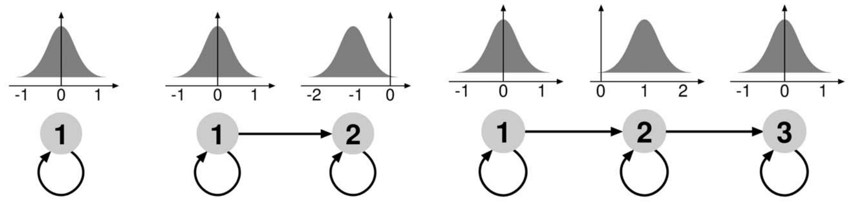


Fig. 1. Three hidden Markov models visualized as directed graphs. The emission pdfs are attached to the states numbered 1 to 3, transitions are shown as directed edges. The models depicted are prototypes for constant (left), down (middle), and up-down-regulation (right).

cannot be used for differentiation or pathogen response experiments. As it is difficult to separate cyclic and noncyclic genes a priori, a method should be able to cope equally well with temporal dependencies indicative of cyclic behavior as well as gene expression time-courses displaying nonperiodic behavior.

The general idea of combining complex statistical models which can reflect temporal dependencies, in particular, HMMs, in cluster ensembles or mixtures has been applied to a number of other problem domains [13], [14], [15], [16], [17], [18].

1.2 Partially-Supervised Learning

Unsupervised learning is not the most appropriate framework for analyzing gene expression data for two reasons. First, usually some relevant information about the genes we want to group will be at our disposal. This might be the function of some gene or the mechanism of regulation of another. If we can trust this information, then we should exploit it in the computation of the grouping, even if it only indicates whether two genes should be members of the same or of distinct groups. Second, many of the approaches used in analyzing gene expression data rely on likelihood maximization with some variant of the locally convergent Expectation-Maximization (EM) algorithm. How additional information—so-called labeled data—can help to alleviate the ensuing estimation problems is an active area of research.

The advantage of combining labeled and unlabeled data was first discovered in the context of learning classifiers. In fact, the decrease in classification error is exponential in the proportion of labeled data [19]. Since then, a number of approaches have been developed following the same general idea; for an overview see [20]. Approaches range from classifying text documents by constructing weighted graphs [21], partitioning graphs by min-cuts controlled by labeled examples [22], or inferring the (minimal) submanifold from labeled and unlabeled data and by using the labeled samples for classification on it [23]. Cozman et al. [24] studied how supervised mixtures get corrupted by unlabeled examples, which can also be interpreted as *transductive learning*, as introduced by Vapnik [25]. A more recent work provides a framework for integrating labeled data when learning hidden Markov random fields [26].

1.3 Method Overview

There are four major building blocks to our method: First, we introduce a class of statistical models, hidden Markov models, able to capture qualitative behavior of time-courses while being robust to delays and rate changes. Second, we select an initial collection of models, either manually by experts, randomly, or automatically by using a deterministic Bayesian top-down clustering approach. Third, we estimate a finite mixture model using prior information in a partially supervised setting. Fourth, we infer groups from the mixture with the level of ambiguity, as measured by Entropy, not exceeding a given threshold, and which are synchronous in behavior, if so desired. We motivate and support our methodological choices with simulations. In subsequent sections, we demonstrate the suitability of our method for the analysis of gene expression time-course data on biological and simulated data, and close with a discussion.

2 HMMs for Gene Expression Time-Courses

Hidden Markov models (HMMs), see [27] for an excellent introduction, are a widely popular model class, particularly in biological sequence analysis. Observations, a biological sequence or a time-course of gene expression, are explained by a sequence of internal, unobservable states of the biological system under investigation. The simplifying assumptions are that changes from one state to another can be described by a Markov chain,[1] and that an observation depends only on the state of the system.

Gene expression levels are easily modeled with a univariate Gaussian for each state. However, in the models we propose, states do not have a specific semantic, contrary to other applications [28]. The HMM topology we employ is essentially a linear chain (following [29], [30], see Fig. 1), except for the possible transition from the last to the first state to accommodate cyclic behavior. The only valid interpretation is that a state expresses the *qualitative* assessment of gene expression level, reflecting contiguous regions of a time-course with *similar* levels of expression. We do not assume or require states to indicate specific steps in the underlying regulatory mechanisms. Once more regulatory pathways become available it will be an interesting question to consider whether such an assignment of semantics becomes possible.

Prototypical behavior of time-courses can be readily encoded in simple models (see Fig. 1 for some examples). The models are parameterized by a triple $(a_{ii}, \mu_i, \sigma_i)$ per state where a_{ii} is the self-transition probability of state i and $N(\mu_i, \sigma_i)$ is the univariate Gaussian of state i. The optional transition from the last to the first state is parameterized with a_{n1}. The models are capable of representing time-courses

1. We assume that the Markov chain is time-homogeneous, first order, and has a finite state space.

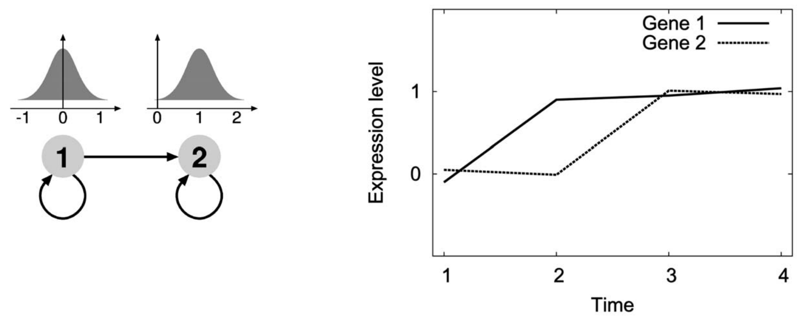


Fig. 2. Capturing asynchronicity: We show how the same model (left) can account for the same qualitative behavior—an up-regulation of gene expression—occurring at different time-points (right). Gene $\#1$ is already up-regulated at the second time-point whereas gene $\#2$ is up-regulated at time-point 3. The corresponding state sequences in the same model are $1, 2, 2, 2$ and $1, 1, 2, 2$ for the two genes, respectively.

which show the same prototypical behavior, such as up-regulation in Fig. 2, occurring *asynchronously*, that is at different time points or with different rates.

If we assume a different state per time-point in the observation—note that this implies $a_{ii} = 0$ for all i—then the linear model is equivalent to a multivariate Gaussian with mean vector $(\mu_1, \ldots, \mu_T)$ and covariance matrix $diag(\sigma_1, \ldots, \sigma_T)$, where T denotes the number of time-points. Typically, the number of states will be substantially smaller than the number of time-points and a simple interpretation of the models in terms of multivariate Gaussians becomes impossible; which state of the HMM explains which time-points is decided *individually* per time-course, so no global covariance structure can exist. In particular, several similar successive measurements will be accounted for by the same state.

Simple as the linear models are, they allow us to capture a wide range of behaviors. It is important to point out that our approach is not limited to such models but rather accommodates arbitrary HMM topologies. As many of the successful applications in time-course modeling [7], [18] show, more complex models, capturing more of the "grammar" observable in the time-courses, which in the case of gene expression is imposed by the regulatory mechanisms, should improve the quality of the results greatly. Also, one can use different emission distributions per state such as the Gamma distribution or finite mixtures of Gaussians.

Missing data is handled in a straightforward manner. A unique symbol is used to represent a missing observation and every state can either emit this symbol with a fixed probability—the proportion of missing observations—or a value distributed according to its Gaussian. We assume that missing data occurs uniformly over time and over the possible models. Note that we do not attempt to estimate or interpolate gene expression levels when they are missing, an undertaking which, given the sparseness of sampling along the time axis, would require, in our opinion, a large degree of optimism. To ensure that all the states are used to account for a time-course, we append a dedicated end-symbol to every time-course, and add a terminal state, not shown in Fig. 1, which solely emits the end-symbol, to each model. This is a routine technique for HMMs.

3 Initial Model Collection

Estimating mixtures is similar to many clustering problems in the sense that the function we strive to optimize has many local optima. A lack of "nice" mathematical properties, such as convexity, precludes us from finding a *global* maximum easily, and the standard algorithms, such as the EM-algorithm, are only guaranteed to arrive at a *local* maximum. Hence, the choice of a starting point becomes crucial, more so because the high dimension of the parameter space describing our models precludes us from performing, say, a grid search for identifying a good starting point. In the following, we introduce three ways of choosing a starting point for the mixture estimation. That is, three ways of choosing for a fixed, given k, an *initial collection* of k HMMs.

3.1 Expert Selection

By using a graphical tool [31], we can verify or falsify the presence of genes exhibiting prototypical behaviors, examples were shown in Section 2, in expression time-courses. If only particular prototypes are of interest, we use corresponding hand-crafted models as the initial model collection. Alternatively, we can create an exhaustive—constant, up, down, up-down-regulation etc.—collection of models encoding all prototypical behaviors. The result of the mixture estimation will quantify the proportion of time-courses adhering to specific prototypes through the mixture weights. In the first case, we can gain some insight as to whether our list of prototypes is a sufficient explanation of the whole data set through a careful analysis of the resulting mixture.

3.2 Randomized Models

We propose randomized initial model collections as follows. Pick a number of states N and create k different N-state models with identical Gaussian emissions centered around zero. Perform Baum-Welch training until convergence with each of the k models. The gene expression time-courses are weighted with random, uniform in $[0, 1]$, weights per model. The resulting randomized model collection (RMC) will explain random subpopulations of the data. In our experience (experiments not shown), the proposed algorithm leads to more reasonable starting points compared to randomizing the mixture parameters

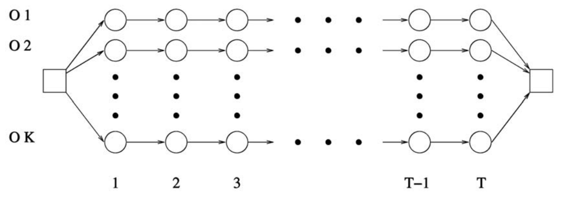


Fig. 3. We show the topologically unconstrained maximum likelihood model with respect to the set of sequences $\{O^1, O^2, \ldots, O^K\}$ of length T. The means of the emission probability distributions are the values of corresponding time points. The variances are set to a fixed constant value.

(cf. Section 4) directly or training k models on a random k-partition of the data.

3.3 Learning Initial Models

The question remains, whether one can propose a more appropriate initial model collection ab initio in a deterministic fashion. Analogous to [13], we first restate the problem by replacing a mixture of k distinct linear HMMs by one larger HMM consisting of a dedicated starting state from which k individual models branch off, so that we can consider learning one HMM instead. Ab initio inference of HMM topology is a challenge, which has been solved only in special cases, through the integration of extensive priors [32], [33], [34] or with extensive computation.

We adapt the Bayesian model-merging algorithm [33] to continuous emissions and linear HMMs. Model-merging starts with the topologically unconstrained maximum likelihood model for a number of sequences, and successively merges states—a merge is known as identification of vertices in graph theory—until a (local) maximization of the posterior is reached. The choice of prior is prudent since it determines when the merging procedure terminates. For large amounts of data, the procedure becomes computationally infeasible as all pairs of states have be to evaluated as merge candidates based on the likelihood of the resulting model.

In our setting, we only consider a simpler problem, as we take advantage of the linear topology of our HMM and further heuristics. We start with a topologically unconstrained maximum likelihood model (see Fig. 3) which has one linear path of states, or branch, per gene-expression time-course, and one state per time-point in each branch. The merging procedure consists of two steps. First, we merge states within the branches. We identify per branch *successive* states whose merging decreases the likelihood the least. As we assume that variances are equal for all states, they only have a scaling effect and we can neglect them in the following. Hence, we only have to identify those successive states whose means are closest. This "horizontal" merging is performed until we arrive at the desired number of states, typical values range from a third to half the number of time-points.

Second, we merge the shrunken branches such that the loss of likelihood is minimal in each merging step with hierarchical clustering. The distance between two models λ_i and λ_j is computed with a modification of the probabilistic distance function [35],

$$D(\lambda_i, \lambda_j) := \frac{1}{T} \log \frac{\mathbf{P}[O^i|\lambda_i]}{\mathbf{P}[O^i|\lambda_j]},$$

where O^i is a random sequence of length T generated by λ_i. Computation of this distance function requires the generation of one (or several) sequences and the computation of likelihoods.

Due to the linear topology, we can compute the distance function directly from the parameters, if we make the additional assumption that the generating state path is independent of the actual emissions. We then can approximate the likelihood along the state path we obtain when we stay the expected duration as determined by the self transition probability in each state. We approximate the distance, neglecting the scaling term, as

$$\bar{D}(\lambda_i, \lambda_j) := \bar{D}_\mu(\lambda_i, \lambda_j) + \bar{D}_a(\lambda_i, \lambda_j),$$

where we define

$$\bar{D}_\mu(\lambda_i, \lambda_j) := \sum_{k=1}^{N} \log \left(\frac{\mathbf{P}[\mu_k^{(i)}|\text{ state } k \text{ in } \lambda_i]^{d_i(k)}}{\mathbf{P}[\mu_k^{(i)}|\text{ state } k \text{ in } \lambda_j]^{d_i(k)}} \right),$$

and

$$\bar{D}_a(\lambda_i, \lambda_j) := \sum_{k=1}^{N} \log \left(\frac{(a_{kk}^{(i)})^{d_i(k)-1}(1 - a_{kk}^{(i)})}{(a_{kk}^{(j)})^{d_i(k)-1}(1 - a_{kk}^{(j)})} \right),$$

N denotes the number of states and $d_i(k) = a_{kk}^{(i)}/(1 - a_{kk}^{(i)})$ is the expected duration of state k in model i. As model λ_i may be the result of previous merges and the decrease of likelihood due to a merge is proportional to the number of time-courses that are affected, we introduce the factor N_i, which is the number of time-courses which have been merged into model λ_i. Note that $\bar{D}_\mu(\lambda_i, \lambda_j)$ accounts for distance due to differences in emissions and $\bar{D}_a(\lambda_i, \lambda_j)$ measures distance due to differences in transition probabilities. We can thus introduce a weighted variant of $\bar{D}(\lambda_i, \lambda_j)$, $\bar{D} = N_i(\bar{D}_\mu + w_{sync}\bar{D}_a)$. This provides a means of driving models toward more synchronicity as differences in transition probabilities get punished more heavily when increasing w_{sync}. Note that D_μ turns out to be the sum of squares of differences of means, weighted by the expectations $d_i(k)$. We use the symmetrized version $\bar{D}_{sym}(\lambda_i, \lambda_j) = \frac{1}{2}\left(\bar{D}(\lambda_i, \lambda_j) + \bar{D}(\lambda_j, \lambda_i)\right)$ in order to identify the pair of models whose merge results in the minimal loss of likelihood in each step. The merge, λ_m denotes the model obtained from merging λ_i and λ_j, is performed by merging corresponding states and computing weighted averages for emissions and durations:

$$\mu_k^{(m)} = \frac{N_i \cdot d_i(k)\mu_k^{(i)} + N_j \cdot d_j(k)\mu_k^{(j)}}{N_i \cdot d_j(k) + N_j \cdot d_j(k)},$$

and

$$d_m(k) = \frac{N_i \cdot d_i(k) + N_j \cdot d_j(k)}{N_i + N_j}.$$

Merging continues until the desired number of models has been reached. Combining the parts yields the following algorithm.

```
Algorithm Bayesian Model Collection (BMC)
Input: A set of gene expression time-courses $\{O^1, \dots, O^K\}$ of length $T$, a desired number of models $K_0$ and a desired number of hidden states $N_0$.
  1. For each $O^i$ define a linear model $\lambda_i$ with $a^{(i)}_{kk} = 0$ and $\mu^{(i)}_k = O^i_k$. Let $N^* = T$ and $K^* = K$.
  2. While $N^* > N_0$ do horizontal merging within each model $\lambda_i$
     (a) Choose
           $k^* = \underset{k \in \{1,\dots,N^*-1\}}{argmin} |\mu^{(i)}_k - \mu^{(i)}_{k+1}|.$
     (b) Merge states $k^*$ and $k^*+1$ into a new state $m$ for which
           $\mu^{(i)}_m = \frac{d_i(k^*)\mu^{(i)}_{k^*} + d_i(k^*+1)\mu^{(i)}_{k^*+1}}{d_i(k^*) + d_i(k^*+1)}$
         and $d_i(m) = d_i(k^*) + d_i(k^*+1)$. Decrease $N^*$ by one.
  3. While $K^* > K_0$ do vertical merging
     (a) Choose
           $(k^*, l^*) = \underset{(k,l), k \neq l}{argmin} \bar{D}_{symm}(M_k, M_l),$
         where $(k, l) \in \{1, \dots, K^*\} \times \{1, \dots, K^*\}$.
     (b) Merge the corresponding models as defined above. Decrease $K^*$ by one.
Output: A set of $K_0$ initial models
```

4 A Mixture of HMMs

Even neglecting the high experimental error rates, biology gives us no reason to believe that genes can be assigned unambiguously to nonoverlapping groups of unique functional category due to the high complexity of interacting networks and the various, context-specific functions of genes. Mixture estimation—a nonstatistical analogue is fuzzy clustering [36]—circumvents this dilemma, which will lead most clustering methods astray. We combine K linear HMMs $\lambda_1, \dots, \lambda_K$ to one probability density function (pdf) for a gene expression time-course by use of a convex combination of the K component probability density functions induced by the HMMs, denoted $p_j(\cdot, \lambda_j)$. The mixture pdf is parameterized by $\Theta = (\lambda_1, \dots, \lambda_K, (\alpha_1, \dots, \alpha_K))$ and defined as $p(\cdot|\Theta) := \sum_{j=1}^K \alpha_j p_j(\cdot, \lambda_j)$. Note, the nonnegative α_j sum to unity. This is just a usual *mixture model* [37], [38] and the resulting likelihood function can be optimized with the EM-algorithm [32], [39], [40], [41] to compute maximum-likelihood estimates for Θ, or *learning* the mixture.

4.1 Partially Supervised Learning

The EM-algorithm we use to learn mixtures is only guaranteed to converge to a local maximum of the likelihood function. This often results in estimates far away from the globally optimal mixture. While this problem is more pronounced in high-dimensional spaces or for small data sets, it also persists in the simplest settings due to identifiability problems [42]. Good initial model collections (Section 3) can alleviate the problem somewhat. A more general idea for improving performance is introduced in the following.

Analogous to [43], we propose *partially supervised learning* as an extension to the EM-algorithm. The training can benefit from prior knowledge about genes, for example, when it is known that they are regulated by the same regulatory pathway. The payoff of even very small quantities, one percent or less, of labels is already large. The robustness of the estimation process with respect to noise increases as well as the quality of the local optimum to which the mixture likelihood converges. In the following, we will argue why the modified EM-algorithm still converges in the case of partially supervised learning.

To apply the EM-algorithm, one assumes the existence of unobservable (or hidden) data $Y = \{y_i\}$, which indicates which component has produced each O^i in the set of time-courses $\mathbf{O}$. Thus, we can formulate a complete-data log-likelihood function $\log L(\Theta|\mathbf{O}, Y)$.

If we are given labeled time-courses, we do not have to guess the corresponding y_i. While the labels do not reveal the *parameters* of the mixture component, they do indicate whether two labeled time-courses have been created by the *same* or by *distinct* components. We denote the set of labeled time-courses with $\mathbf{O}_L$ and the set of unlabeled ones with $\mathbf{O}_U$. For a time-course O^i from $\mathbf{O}_L$, we set the value of y_i to its component label l_i and maintain this assignment throughout the running time by setting $\mathbf{P}[\lambda_j|O^i] = 1$ for $j = l_i$ and zero else. This can be thought of as conditioning the relevant distributions and the likelihood on the known labels, yielding a Q-function (cf. [41]; the Θ^t are the estimates for the maximum likelihood in the t-th iteration), which splits into two sums,

$$Q(\Theta, \Theta^t) := \sum_{O^i \in \mathbf{O}_L} \log\big(\alpha_{l_i} p_{l_i}(O^i|\lambda_{l_i})\big) + \sum_{O^i \in \mathbf{O}_U} \sum_{j=1}^K \log\big(\alpha_j p_j(O^i|\lambda_j)\big) \mathbf{P}[j|\Theta^t, O^i],$$

and for which the usual local convergence result holds.

4.2 Estimating the Number of Components

Estimation of the correct or optimal number of components in a mixture is an unsolved problem. When we make use of our prior beliefs in modeling, we can tackle this task in a Bayesian framework, by comparing the competing mixture models M_i with Bayes Factors. In other words, calculating the ratio of posteriors without favoring one or the other a priori; i.e., $B_{12} = \mathbf{P}[\mathbf{O}|M_1]/\mathbf{P}[\mathbf{O}|M_2]$, where M_1 and M_2 are two models with respective parameters Θ_1 and Θ_2. Then, it is possible to compare several models at once, rather than two by two as in frequentist statistical tests. When we use the EM-algorithm to estimate maximum likelihood models, approximate Bayes factors can be easily deduced from the Bayesian information criterion (BIC) [44], $-2 \log \mathbf{P}[\mathbf{O}|M_K] \approx -2 \log L(\mathbf{O}|\Theta_K, M_K) + f_K \log n$, where K is the number of components, $L(\mathbf{O}|\Theta_K, M_K)$ is the maximized mixture log-likelihood with K components, f_K is the number of free parameters in Θ_K, and n is the number of sequences in $\mathbf{O}$.

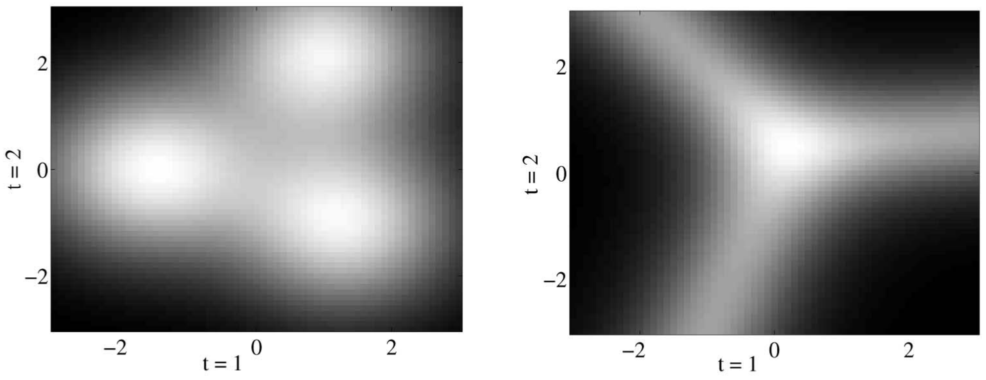


Fig. 4. We depict a three component mixture model as a density plot for a two time-point gene expression time-course on the left. Darker colors correspond to less dense regions. On the right, we depict the entropy of the corresponding points posterior distribution over components. Darker colors correspond to lower entropy.

The right-hand term of the formula represents a factor for penalizing more complex models, since the fit of a model tends to improve as the number of parameters increases. The smaller the value of BIC, the better the model. It has been shown that BIC does not underestimate the number of true components asymptotically and performs well in simulation studies [38]. Each state of an HMM is parameterized by the triple $(a_{ii}, \mu_i, \sigma_i)$, see Section 2. As a result, the number of free parameters in a model λ_j is $f_{\lambda_j} = 3N_j$, respectively, $3N_j + 1$ for models with loops from last to first state, where N_j is the number of states in λ_j. Hence, the number of free parameters in Θ_K is defined as $f_K = \sum_{j=1}^{K} (f_{\lambda_j} + 1) - 1$.

5 Inferring Groups

While the mixture encodes relevant information about the data, groups are easier to handle and give a more accessible starting point for further analysis. The simplest way of *decoding* a mixture, that is, to infer groups in the data, is to interpret the mixture components as descriptive models of non-overlapping clusters and assign each time-course O to the cluster j of maximal posterior $\mathbf{P}[\lambda_j|O]$. This is exactly the assignment step in model-based or k-means clustering. However, a mixture encodes much more information. Inspection of the discrete distribution $d(O) := \{\mathbf{P}[\lambda_i|O]\}_{1\leq i\leq K}$, the posterior of components given a time-course, reveals the level of ambiguity in making the assignment. This ambiguity can be quantified (see Fig. 4) by computing the entropy $H(d(O))$, where $H(p) : = -\sum p_i \log \frac{1}{p_i}$ is the usual Shannon entropy. Choosing a threshold on the entropy yields a grouping of the data into at most $K+1$ groups. If $H(d(O))$ is below the chosen threshold, we assign O to the component with maximal posterior as above. Otherwise, O is assigned to the $(K+1)$st group, which contains all genes which cannot be assigned unambiguously.

There is no simple way to choose an optimal threshold on the entropy. However, an interactive graphical user interface [31], displaying the time-courses and their assignment to clusters when the user changes the threshold provides one exploratory way to settle on a value. While a threshold on the entropy clearly helps in identifying unambiguous groups, we will demonstrate by use of gene annotation information that the groups become more biologically meaningful as the entropy threshold decreases.

5.1 Entropy Threshold and Specificity of Gene Annotation

Gene Ontology (GO) describes genes in three distinct categories [45]: cellular component, molecular function, and biological process. Such an ontology has the form of a directed acyclic graph (DAG), where the leaves are genes and the internal nodes are terms (or annotations) describing gene function, gene cellular localization, or the biological processes genes take part in. Leaves near the root describe very general processes, while nodes near the leaves describe specific ones. One should expect that the more unambiguous a cluster is, the more specific information it contains. Following this rationale, we evaluated the relationship between ambiguity of gene clusters and the specificity (or level) of GO annotations.

In order to find GO annotations related to a given group of genes, one should look for annotation terms that are overrepresented in this group. The probability that this overrepresentation is not found by chance can be measured with the use of a hype-geometric Fisher exact test [46]. Let n be the total number of annotated genes in GO (reference group), and m be the number of genes annotated with a specific GO term. This will give us m positive genes and $n-m$ negative genes. If we draw k genes from the reference group (or analogously obtain a cluster with k genes), we obtain q positive genes and $k-q$ negative genes, see Table 1 for a two-by-two contingency table representation of these terms. We are interested in observing how unusually large this value q is, given n, m, and k. This can be achieved by computing a p-value defined by $\mathbf{P}[X \geq q]$, where X is defined by $\{\mathbf{P}[x=i]\}_{1\leq i\leq k}$, and

$$\mathbf{P}[x=i] = \frac{\binom{m}{i}\binom{n-m}{k-i}}{\binom{n}{k}}.$$

Because of the effects of multiple testing, a subsequent correction of the p-values is necessary. For example, if we have 1,000 GO terms, and a p-value of 0.1 is used, at least 100 false positives are expected. To correct this, we apply the false positive discovery rate proposed in [47]. For

TABLE 1
Two-by-Two Contingency Table for Genes Annotated or Not Annotated by a Given GO Term

	Positive Genes	Negative Genes	Total
In Cluster	q	$k-q$	k
Not In Cluster	$m-q$	$(n-k)-(m-q)$	$n-k$
Total	m	$n-m$	n

further details, see [48], where GOStat, the tool used in this work, is described.

The calculation of the specificity (or level) of the annotations from a set of genes is straightforward. Given a cluster, we repeat the above test for each term in GO, and retrieve the ones which do not exceed a given p-value. Then, the length of the path from the root to each enriched term is computed. Since GO is a DAG, one node can be reached by more than one path from the root, so the average of all possible path lengths is taken.

The results show that the level peaks at distinct thresholds for each cluster (Fig. 8), suggesting that distinct threshold should be used. This, however, does not prevent us from use of the average specificity level for all clusters from a given data set to quantify the overall behavior of the GO level as a function of the threshold.

5.2 Viterbi Decomposition

The groups we infer are asynchronous by design (cf. Section 2). If synchronous subgroups are required, they can be obtained by use of the Viterbi decomposition. The sequence of HMM states which generated each particular time-course may vary within a group. The task now is to distinguish between sequences whose generation procedure has been decisively different and thus to infer subgroups of time-courses following synchronous generation mechanisms.

This translates to finding the path through the hidden states of the models most likely to have been used to generate each sequence, the so called Viterbi paths. Our approach is based on the assumption that there is one hidden state that defines the characteristic pattern of the time-courses assigned to each model. This is, according to our experience, reasonable. Take, for example, a group of cyclic time-courses. Sorting these according to the time at which the first peak is reached usually produces subgroups of time-courses with the same phase shift. Nonetheless, our approach can easily be extended to check for more than one state if needed.

```
Algorithm Viterbi decomposition
Input: An HMM $\lambda$ and a corresponding group of gene
        expression time courses $\mathbf{O}=\{O^1,\dots,O^K\}$ of length $T$.
  1. For each $i \in \{1,\dots,K\}$ compute the Viterbi path
     $v_i := v_i(1)v_i(2)\dots v_i(T) \in Q^T$ of time-course $O^i$, where
     $Q=\{q_1,\dots,q_N\}$ is the set of all hidden states of $\lambda$.
  2. For each $q \in Q$ partition the set $\mathbf{O}$ into a subgrouping
     $\mathcal{O}(q)=\{O(q)_1,\dots,O(q)_{j_q}\}$, such that for $j \in \{1,\dots,j_q\}$

        $O^l, O^m \in O(q)_j :\Leftrightarrow T(q)_l = T(q)_m =: T(q)_j$,

     where $T(q)_l = \{t \in \{1,\dots,T\} \,:\, v_l(t)=q\}$ is the set of
     time-points $t$, where gene $O^l$ is in state $q$.
  3. For each subgrouping $\mathcal{O}(q)$ compute a coefficient $C(q)$
     assessing the quality of the subgrouping (see below).
  4. Choose the hidden state $q^*$, s.t.

        $q^* = \underset{q}{argmax} \quad C(q).$

  5. Within the subgrouping $\mathcal{O}(q^*)$ iteratively join pairs of
     subgroups of time-courses yielding the largest
     increase of $C(q^*)$. Stop if no joining of two subgroups
     results in an increase any more and return the
     obtained subgrouping (comments see below).
```

In the following, we will elaborate on some details of the Viterbi decomposition.

Step 3: The coefficient for the subgrouping $\mathcal{O}(q)$ can be written as $C(q)=b(q)-a(q)$, with $b(q)=1/N_b \sum_{k,l} D_b^q(O^k,O^l)$, where (k,l) ranges over all pairs of sequences (O^k,O^l) such that O^k and O^l are contained in different subgroups, N_b is the total number of such pairings and $D_b^q(O^k,O^l)=\frac{1}{|T_{k,l}|}\sum_{t\in T_{k,l}}(O^k(t)-O^l(t))^2$, where $T_{k,l}=\{t\in\{1,\ \dots,T\} \mid (v_k(t)=q \wedge v_l(t)\neq q) \vee (v_k(t)\neq q \wedge v_l(t)=q)\}$ (i.e., all time-points where exactly one of the two genes is in state q) and analogously $a(i)=1/N_a\sum_{k,l}D_a^q(O^k,O^l)$, where now (k,l) ranges over all tuples, which correspond to sequences O^k, O^l, which can be found in the same subgroup (N_a is the total number of such pairings) and $D_a^q(O^k,O^l)=\frac{1}{|T_{k,l}|}\sum_{t\in T_{k,l}}(O^k(t)-O^l(t))^2$, where now $T_{k,l}=\{t\in\{1,\dots,T\} \mid (v_k(t)=q \wedge v_l(t)=q)\}$. Note that $C(q)$ can be understood as a local silhouette coefficient.

Step 5: If two subgroups $O(q)_{i,j} := O(q)_i \cup O(q)_j$ are joined and $T(q)_i, T(q)_j$ are the corresponding sets of time points (obtained through the defining property of these subgroups, see Step 2) then $\forall O^k \in O(q)_{i,j} : T(q)_k := T(q)_i \cup T(q)_j$, which can be understood as an updating of Viterbi paths.

Step 5 can be seen as a hierarchical clustering procedure. It is motivated through the fact that sometimes different subgroups are essentially synchronous although their Viterbi paths slightly differ even in regions where significant changes occur. In Fig. 6, we show results of the Viterbi decomposition on biological data.

6 Evaluation

It is still open to debate what constitutes appropriate evaluation of methods for gene expression time-course analysis. Regulation is not well understood and the annotation available is too sparse and too inconsistent to support a large-scale evaluation. Hence, biological data can only provide anecdotal evidence, as demonstrated by prior work. We used HeLa cell-cycle and Yeast sporulation data, and a fully annotated subset [11] of Yeast cell-cycle data. Note, this data set contains only a small percentage of the original genes and the selection criteria is likely to introduce a bias. To test our method under more controlled conditions and for benchmarking, we resorted to artificial data. All data sets, links to software and results are available in the supplementary material (http://algorithmics.molgen.mpg.de/ExpAna/).

6.1 Data

6.1.1 Yeast Cell Cycle (Y5)

This data set represents the expression levels of more than 6,000 genes during two cell cycles from Yeast measured in 17 time points [50]. Following [50], we used a subset, 5-phase criterion, abbreviated Y5, of 384 genes visually identified to peak at five distinct time points [50], each representing a distinct phase of cell cycle (Early G1, Late G1, S, G2, and M). All the genes in Y5 are annotated. The expression values of each gene were standardized, which can enhance the performance of model-based clustering methods, when the original data consists of intensity levels [11].

6.1.2 Yeast Sporulation (YSPOR)

This data set [51] contains gene expression measurements during sporulation for more than 6,400 genes of budding yeast. The measurements were taken at seven time points (0h, 0.5h, 2h, 5h, 7h, 9h, and 11h). Clones with more than 20 percent of values missing were excluded. The data was preprocessed by extracting all those genes with an absolute fold change of at least two in at least one time point. The resulting data set contains 1,171 genes.

6.1.3 HeLa Cell Cycle (HeLa)

We used published data from a time-course experiment [52], in which the authors measured genome wide gene expression of synchronized HeLa cells. We used the raw data from doubly thymidine experiment three as provided by the authors in the supplementary information. In this data set, HeLa cells, which have been arrested in S phase by a double thymidine block, were measured every hour from 0 to 46 hours. For reasons of comparison, we excluded clones with missing values from further analysis. The data was also filtered by extracting all genes that do not show a two-fold change as in YSPOR. This resulted in a data set containing 2,272 expression time courses as logarithms of ratios with respect to a control. Additionally, we used a list of genes the regulation of which has been described in the literature to depend on the cell cycle [52].

6.1.4 Simulated Data 1 (SIM1)

To create an unbiased benchmarking data set, we chose to make only mild assumptions, *independent* from the underlying assumptions of our method and those we compare against about the nature of the data, while still reflecting the realities of microarray experiments. We assume three broad categories of genes, cell-cycle regulated, noncell-cycle regulated, and unregulated genes. We chose sine functions as a "true" model for the first, linear functions for the second and $const = 0$ for the third category (see [30] for details). Randomization is performed by changing phase, frequency and amplitude, shifting all values and adding Gaussian noise.

6.1.5 Simulated Data 2 (SIM2)

We selected eight HMMs encoding the possible three-segment regulation behaviors (e.g., down-down-down, up-down-down) and used a Monte-Carlo algorithm (variances of emission probabilities were set to 0.2) to generate 100 time-courses from each of the eight HMMs.

6.2 Measure of Agreement

External indices are used to assess the degree of agreement between two partitions, say, between a clustering U and classes V obtained from independent category labels [53]. Among a number of existing indices, the use of corrected Rand (CR) is suggested by [54].

Let $U = \{u_1, \dots, u_r, \dots, u_R\}$ be the partition given by the clustering solution, and $V = \{v_1, \dots, v_c, \dots, v_C\}$ be the partition defined by the a priori classification. The corrected Rand is defined as

$$\frac{\sum_{i=1}^{R}\sum_{j=1}^{C}\binom{n_{ij}}{2} - \binom{n}{2}^{-1}\sum_{i=1}^{R}\binom{n_{i\star}}{2}\sum_{j=1}^{C}\binom{n_{\star j}}{2}}{\frac{1}{2}[\sum_{i=1}^{R}\binom{n_{i\star}}{2} + \sum_{j=1}^{C}\binom{n_{\star j}}{2}] - \binom{n}{2}^{-1}\sum_{i=1}^{R}\binom{n_{i\star}}{2}\sum_{j=1}^{C}\binom{n_{\star j}}{2}},$$

where n_{ij} represents the number of objects in clusters u_i and v_j, $n_{i\star}$ indicates the number of objects in cluster u_i, $n_{\star j}$ indicates the number of objects in cluster v_j, and n is the total number of objects. CR can take values in $[-1, 1]$, where the value 1 indicates a perfect agreement between the partitions, whereas values below some $\epsilon > 0$ correspond to cluster agreement found by chance (see [54] for simulation studies).

We also use sensitivity, $\frac{\#TP}{\#TP+\#FN}$, and specificity, $\frac{\#TP}{\#TP+\#FP}$, where, for a given U and V, TP denotes the number of pairs of objects in the same cluster in U and same class in V. The remaining three types of pairs are counted as FP (same cluster, distinct class), TN (distinct cluster and class) and FN (distinct cluster, same class). The main distinction between CR and these two indices is the fact that CR weighs both types of error (FP and FN) equally. As a consequence, the use of sensitivity and specificity is complementary to the use of CR.

6.3 Results on Biological Data

We used both randomized model collection (RMC) and Bayesian model collection (BMC) to obtain starting points for mixture estimation. In each model collection, we added a designated *noise*-component, which is simply a one-state model with $N(0, \sigma)$-emissions for a large value of σ, exempted from training. This mixture component accounts for time-courses which do not fit any of the other components and, thus, avoids unnecessary "broadening" of the other components. We used BIC in order to select the number of components. We repeated the experiments 30 times for varying numbers of k and used BIC to choose a best k. Best performance is shown.

6.3.1 HeLa

Cell cycle regulators as, for example, different cyclins, E2F, PCNA and HDAC3 are known to be active in different stages in the cell cycle. Furthermore, they regulate each other, either directly (e.g., E2F1 acting on CyclinD) or indirectly (E2F1 regulates p27 and vice versa via CDK2-CyclinA). Thus, one expects to find these patterns of regulatory activity in the underlying gene expression data set. CyclinB and CyclinA both act while being bound to CDC2 during the transition from G2 to M Phase. They are coordinately regulated and there is a clear phase shift compared to E2F1, which is active in the transition from G1 phase to S phase. CyclinF is known to have a sequence similar to CyclinA and B, but its function is largely

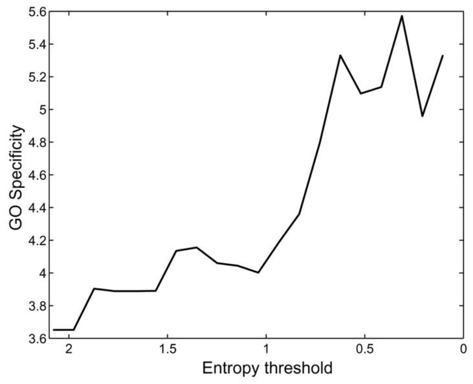


Fig. 5. We show the mean GO annotation level of all groups from HeLa versus the entropy threshold. The lower the threshold, the more unambiguous is a cluster.

unknown [55]. It is apparently regulated synchronously to CyclinA and B [30]. On the other hand, E2F1 regulates p27 which is demonstrated by a clear phase shift in the time-courses. PCNA is also needed for the initiation of S phase and is regulated clearly in G1/S phase. HDAC3 which is needed to permit access to the DNA and, thus, allow transcriptional regulation is expressed during G1/S transition as well.

The initial collection consisted of 35 24-state models obtained from RMC. We used the five G2/M phase genes described above as labels for one cluster and four genes of the G1/S phase for a second one. Decoding the mixture resulted in two groups containing the labeled time-courses of size 91 and 14, respectively (Fig. 7). We computed a Viterbi-decomposition of the larger group. The first subgroup contained 26 genes known to be G2 and one G2/M, the second 11 G2 and 19 G2/M, the third 31 G2/M, two M/G1 and 1 G1/S. The second group (not shown) contained twelve G1/S and two S-phase genes. Both CDC2 representatives are found in the same subgroup (Fig. 7, phase 1). Furthermore, cyclin A (Fig. 7, phase 2) and cyclin B (Fig. 7, phase 3) are assigned to different subgroups, shifted in phase with respect to the one containing CDC2. Moreover, all time-courses that are assigned to the different phases of our G2, G2/M phase cluster are known to be cell cycle regulated in their respective phase [52]. The same holds for the G1/S, S phase subgroups. Apparently, the modest amount of prior information helped find highly specific groups of synchronously expressed genes.

6.3.2 Specificity of Gene Annotation

We only used the complete biological data sets (HeLa and YSPOR) for experiments relating entropy threshold and GO specificity. Given the best performance model, g threshold values from $\max H(d(O^i))$ to zero were selected and their corresponding clusters analyzed (the value zero is not included since it yields no groups). Only GO terms with a $p\text{-value}$ below 0.1 were considered.

In both HeLa and YSPOR, we observe an increase in GO specificity until a certain low threshold value, followed by a decrease of specificity (see Fig. 5). The mean level raises considerably (around 2.0) until it reaches the threshold value 0.3 with HeLa, while in the YSPOR an increase of 0.5 was found.

In individual clusters distinct behaviors can be noticed (Fig. 8a). In cluster one from YSPOR, at the maximum threshold, 24 genes are related to the term "M Phase." After applying a threshold of 0.41, 18 of these genes were still present in the cluster. Cluster four, for instance, has no enriched terms for high thresholds, but for values around 0.6 and 0.4, enrichment for terms related to "ATP" and "nucleus" is found. On HeLa (Fig. 8b), we observe distinct GO specificity patterns (we display only six representative groups out of the 35). Cluster one, which has a high and increasing GO level, has 12 out of 37 genes associated with the GO term "mitotic cell cycle," and these 12 genes are still in cluster one after the application of the lowest threshold (see Table 2). Cluster three presents no enriched term for high cutoff values, but for values around 0.7 to 0.3, GO terms related to "ion binding" present enrichment. A similar behavior is found in cluster six, where term enrichment is only present with values threshold values lower than 1.5. Cluster five, where the GO level decreases after the value 1.1, does not contain more than two genes

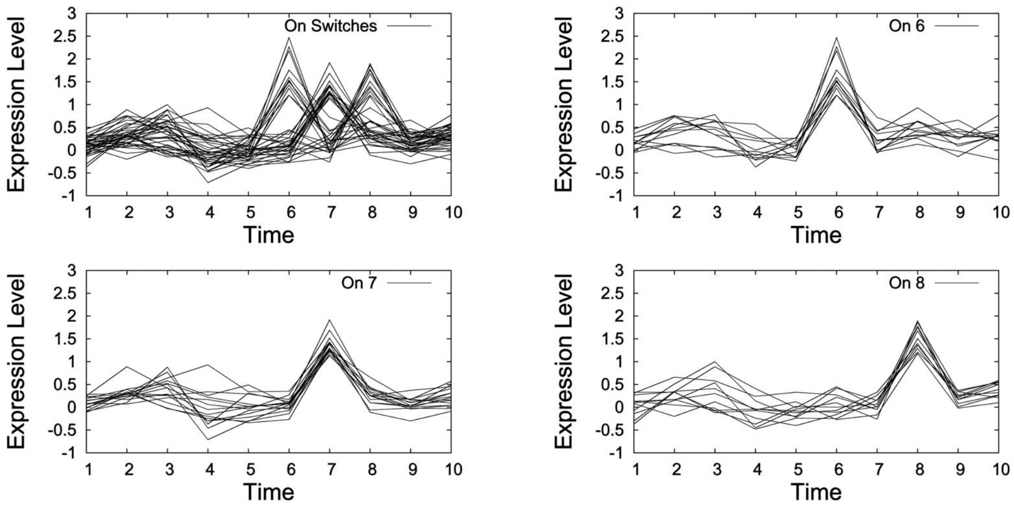


Fig. 6. A cluster of gene expression time-courses from the Fibroblasts data set [49] (see [29] for an analysis) displays "on-switch" behavior at different time-points decomposed by the Viterbi Decomposition. The top left picture shows the complete ensemble of "on-switch" genes whereas the other plots show the synchronous components.

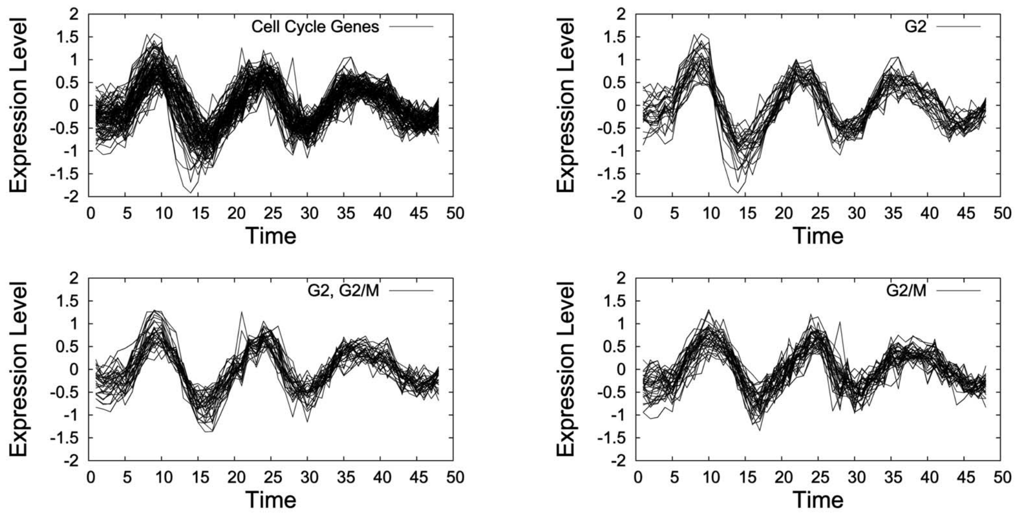


Fig. 7. We computed a mixture model from nine labeled and 2,263 unlabeled time-courses for the HeLa data set. One group (top left) was seeded with five of the labeled time-courses. It was subsequently decomposed into three subgroups, corresponding to groups of synchronous genes, with the Viterbi decomposition. The first subgroup contains mainly G2 genes (phase 1, top right), the second G2 as well as G2/M genes (phase 2, bottom left) and the third mostly G2/M genes (phase 3, bottom right).

related to the same term for any of the thresholds values analyzed. There are two possible explanations for the decreasing behavior of cluster five. Either this is a cluster of poor quality or, most probably, it contains genes, which are not annotated in GO. In contrast to Yeast, where the majority of genes are annotated, only parts of the Human genome are present in GO. Furthermore, this scarce annotation is possibly biased toward well-known biological processes. This difference of GO coverage between species can also be the reason why Yeast groups have higher GO specificity levels. This, however, does not invalidate the results shown above, as they do display a increase in the majority of clusters from the YSPOR data set and in most of the clusters from HeLa.

6.4 Benchmarking Results

Three methods of model initialization were used: the randomized model collection (RMC), the Bayesian model collection (BMC), and k-means initialization (KMI), to supply a base line. In the latter, a standard k-means algorithm was first applied to the data, and the k groups obtained were used to initialize models. In addition to mixture estimation, we also evaluated HMM-clustering [29], and partially supervised mixture estimation. When needed, Viterbi decomposition was applied to the results (see Section 6.8.7). Experiments using RMC and KMI were replicated 50 times (mean results are shown). Note, BMC does not require repetitions, as it is a deterministic algorithm. We also evaluate the performance of Caged [9], Splines [8] (as obtained from the authors using default parameters) and k-means [56].

6.4.1 Y5

The k-means algorithm obtained a remarkably good result with a CR of 0.43 (0.56 specificity and 0.56 sensitivity). Mixture estimation with BMC and a posterior Viterbi decomposition also performed quite well, with a CR of 0.467, 0.55 specificity, and 0.66 sensitivity. The best results (see Table 3) were obtained by the partially-supervised mixture estimation with only 15 labels followed by VD (0.49 CR, 0.56 specificity and

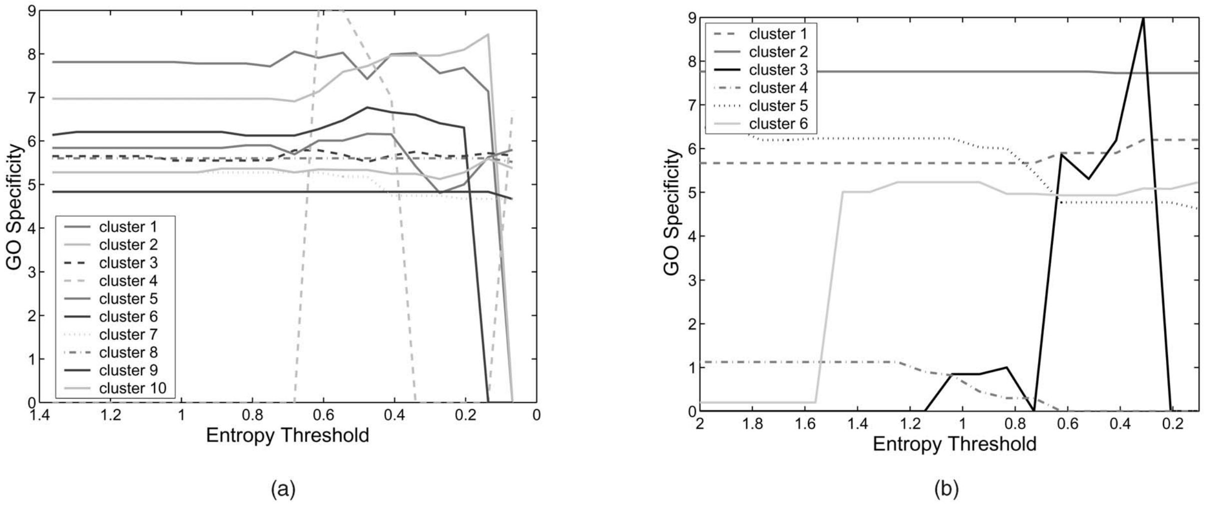


Fig. 8. (a) We present the GO specificity levels of 10 individual clusters from the YSPOR and (b) the levels of six out of the 35 clusters from HeLa.

TABLE 2
Ten Most Enriched GO Terms of Clusters One and Five from Hela

GO number	Counts	p-value	GO Term
GO:0000278	12 of 203	1.05e-17	mitotic cell cycle
GO:0000279	10 of 181	2.68e-14	M phase
GO:0007067	9 of 132	1.08e-13	mitosis
GO:0000087	9 of 138	1.22e-13	M phase of mitotic cell cycle
GO:0000280	9 of 168	5.93e-13	nuclear division
GO:0007049	12 of 833	4.06e-11	cell cycle
GO:0008283	12 of 1180	2.04e-09	cell proliferation
GO:0000074	8 of 424	5.77e-08	regulation of cell cycle
GO:0005819	4 of 51	2.94e-06	spindle
GO:0015630	6 of 278	3e-06	microtubule cytoskeleton
GO:0000278	12 of 203	6.22e-19	mitotic cell cycle
GO:0000279	10 of 181	3.11e-15	biological M phase
GO:0007067	9 of 132	1.66e-14	mitosis
GO:0000087	9 of 138	1.87e-14	M phase of mitotic cell cycle
GO:000028	9 of 168	9.14e-14	nuclear division
GO:0007049	12 of 833	2.6e-12	cell cycle
GO:0008283	12 of 1180	1.36e-10	cell proliferation
GO:0000074	8 of 424	1.19e-08	regulation of cell cycle
GO:0015630	6 of 278	1.08e-06	microtubule cytoskeleton
GO:0005819	4 of 51	1.24e-06	spindle
GO:0046579	1 of 3	0.0605	positive regulation of RAS protein signal transduction
GO:0051057	1 of 3	0.0605	positive regulation of small GTPase mediated signal transduction
GO:0046578	1 of 4	0.0605	regulation of RAS protein signal transduction
GO:0008168	2 of 161	0.0605	methyltransferase activity
GO:0030742	1 of 6	0.0605	GTP-dependent protein binding
GO:0051056	1 of 6	0.0605	regulation of small GTPase mediated signal transduction
GO:0006379	1 of 6	0.0605	mRNA cleavage
GO:0005522	1 of 6	0.0605	profilin binding
GO:0016741	2 of 167	0.0605	transferase activity, transferring one-carbon groups
GO:0006378	1 of 7	0.0605	mRNA polyadenylylation
GO:0008168	2 of 161	0.000432	methyltransferase activity
GO:0016741	2 of 167	0.000432	transferase activity, transferring one-carbon groups
GO:0016742	1 of 8	0.00318	hydroxymethyl-, formyl- and related transferase activity
GO:0004047	1 of 9	0.00318	aminomethyltransferase activity
GO:0004372	1 of 12	0.00339	glycine hydroxymethyltransferase activity
GO:0016491	2 of 954	0.00472	oxidoreductase activity
GO:0007283	1 of 130	0.0532	spermatogenesis
GO:0048232	1 of 130	0.0532	male gamete generation
GO:0007276	1 of 161	0.0576	gametogenesis

We depict the terms enriched for cluster one, threshold values 1.98 and 0.31 and cluster five, same thresholds (top to bottom). Counts represents the terms q and $m - q$, respectively. Cluster one shows a high enrichment with terms associated with "mitotic cell cycle," even when the lowest threshold is applied. In cluster five, GO terms are poorly enriched, and the application of the threshold excludes a high number of genes from the cluster, 16 of the original 18 genes (see the supplementary material for the enrichment of other clusters).

0.68 sensitivity). The results of Caged were not included, since it could only find one cluster (this cannot be controlled by the user).

Except for the partially supervised estimation, all other methods were not able to separate some genes from groups Late G1 and S or M and Early G1. These phases are separated by a phase-shift and their genes have very similar time courses. The application of the Viterbi Decomposition yielded noticeable improvements in distinguishing genes from these overlapping groups in all methods.

Even though this data set has been useful for comparing distinct methods, it should be pointed out that it does not represent the ideal data for benchmarking. It almost exclusively contains cell cycle genes, which does not represent the typical input, where usually all behaviors of temporal gene expression are present.

6.4.2 SIM1

Two of the more involved methods, Caged [9] and Spline-based clustering [8] (using default parameters) only reached a specificity and CR of less than 0.5 (see Table 4). The main error made by Caged is deciding on too few clusters (this cannot be controlled by the user), which leads to merging of several classes (C1 and C2, respectively, C3-C6 into one cluster). HMMs perform quite well with all initialization methods. In particular, mixture estimation with BMC achieved a high CR of 0.84, over 91 percent specificity and over 82 percent sensitivity. The tests also show very clearly the positive effect of partially supervised learning. It suffices to have labels for twenty time-courses to obtain a specificity and sensitivity exceeding 0.91 and 70 labels to have both values around 0.94.

TABLE 3
Results of the Different Methods on Y5

Description	CR	Spec.	Sens.
HMM Mix. & RMC	0.330	0.488	0.475
HMM Clu. & RMC	0.331	0.474	0.490
Splines	0.362	0.494	0.516
HMM Clu. & KMI	0.380	0.534	0.502
HMM Clu. & BMC	0.388	0.520	0.543
HMM Mix. & BMC	0.390	0.531	0.527
HMM Mix. & KMI	0.391	0.538	0.517
HMM Clu. & KMI &VD	0.407	0.470	0.732
K-means	0.430	0.563	0.557
HMM Mix. & KMI &VD	0.432	0.502	0.718
HMM Clu. & RMC &VD	0.454	0.534	0.672
HMM Mix. & RMC & VD	0.458	0.540	0.664
HMM Clu. & BMC & VD	0.462	0.547	0.654
HMM Mix. & BMC & VD	0.467	0.551	0.658
HMM Mix. 2.5% labeled & VD	0.486	0.559	0.684

6.4.3 Robustness of Mixtures versus Clustering

Besides the biological motivation for choosing mixtures, we would like to argue also from the robustness point of view. We compared the robustness of mixture estimation with HMM-clustering [29] as follows: We selected eight HMMs encoding the possible three-segment regulation behaviors (e.g., down-down-down, up-down-down) and used a Monte-Carlo algorithm to generate twenty sets of 200 time-courses each from the eight HMMs (32,000 artificial time-courses total). Noise was introduced by adding i.i.d. Gaussian variates with mean zero and variance σ^2 for increasing values of σ to each simulated observation. Subsequently, both clustering and mixture estimation were run until convergence, with the true generating models as the initial model collection in both case. To quantify performance, we computed the sum of squared deviation of means between true and estimated models over all states and all models in the collection (see Fig. 9).

The results on SIM1 and Y5 also confirms the advantages of mixture estimation over clustering. In most of the experimental settings, mixture estimation performed as well as or better than clustering.

TABLE 4
Results on SIM1

Method	CR	Specificity	Sensitivity
Splines	0.343	0.584	0.438
Caged	0.404	0.997	0.410
k-means	0.715	0.804	0.757
HMM Mix.& RMC	0.759	0.787	0.848
HMM Clu. & KMI	0.791	0.882	0.794
HMM Clu. & RMC	0.799	0.888	0.801
HMM Mix. & KMI	0.801	0.850	0.847
HMM Clu. & BMC	0.803	0.903	0.788
HMM Mix. & BMC	0.836	0.912	0.829
HMM Mix. 0.6% labeled	0.885	0.911	0.911
HMM Mix. 2.0% labeled	0.927	0.946	0.939

6.4.4 Partially Supervised Learning

We demonstrated the importance of partially supervised learning for biological data in Section 6.2. To give a better understanding, we performed the following experiments on SIM1 and SIM2. We randomly picked a number of labels from each class, from 0 percent to 25 percent of the total number of genes. For each number of labels used, 30 replications were performed. Sensitivity, specificity and CR increase (Fig. 10) as the proportion of labels increases, and even small numbers of labels have a pronounced effect.

6.4.5 Finding the Number of Components

BIC was able to find the correct number of components in SIM1 (six) and underestimated the number of components by only one in Y5 (Fig. 11). This underestimation is another indication of the deficiency of the methods to separate some classes of Y5 (there is no way to integrate VD in the estimation of the number of components). Our results agree with the literature on the performance of BIC on large data sets [37]. Furthermore, the calculation of BIC is very simple and requires no replication of experiments, in contrast to other methods [14].

6.4.6 Comparing Initialization Methods

As shown in the results of SIM1 and YSPOR, BMC obtained higher accuracy than KMI and RMC in most of the experimental settings. Another important characteristic in favor of BMC is its deterministic nature. As a consequence, there is no need to perform replicates of the experiments and to choose the best replicate, contrary to use of RMC or KMI.

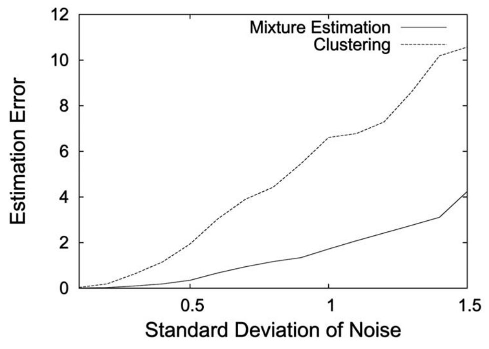


Fig. 9. On the y-axis, squared estimation error summed up over all states and all models and averaged over the 20 samples is plotted against different levels of added noise, $N(0, \sigma), 0.1 <= \sigma <= 1.5$ in increments of 0.1). A Wilcoxon test comparing deviation of estimated model parameters from their true value showed significantly lower estimation error for mixture estimation.

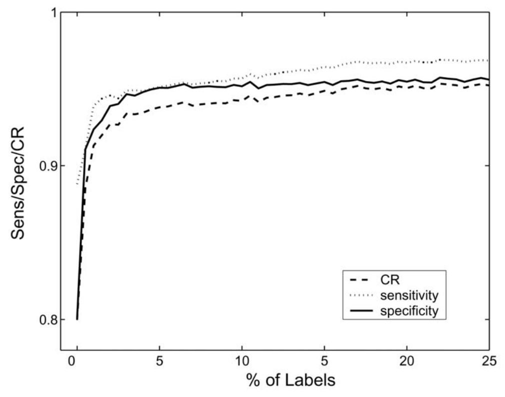


Fig. 10. We depict sensitivity, specificity and CR of a partially supervised clustering procedure depending on the percentage of labeled data in the input on SIM1.

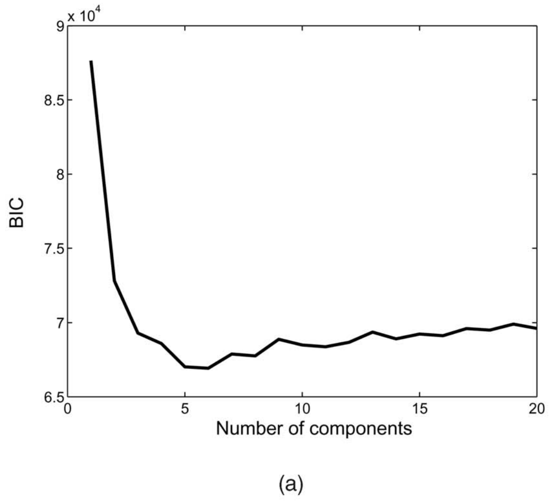
6.4.7 Viterbi Decomposition

The results on Y5 suggest that some clustering methods fail to separate time-course classes with very subtle distinctions. In this particular case, these distinctions are based on small phase-shifts in successive cell cycle phases. This seems to be typical for biological data, and the results obtained in both cell cycle data sets (Y5 and HeLa) show that with VD we can group those genes correctly. The necessity of the application, however, cannot be generalized. For instance, in SIM1, where no such phase-shifted classes exists, the use of VD would not improve the results, since clustering was already able to group the genes correctly.

7 Summary

We present a robust and efficient approach to analyze gene expression time-course data with a mixture of hidden Markov models. The method can easily make use of prior knowledge about genes due to a partially supervised training procedure, which greatly increases robustness and the quality of the local optima found. Availability of such labels is a realistic assumption for the analysis of gene expression time-courses. Simultaneous analysis of cyclic and noncyclic time-courses is possible and neither missing values nor high levels of noise pose a serious problem. Mixtures are, for reasons of the complexity of gene function and regulation, a more appropriate model of biological reality than clusterings. They also have computational advantages, demonstrating a higher robustness to noise compared to clustering. Most importantly, mixtures allow us to quantify the level of ambiguity when assigning genes to groups based on their gene expression time-courses. Our simple, information-theoretic approach supports interactive exploration of assignments at various levels of uncertainty, as well as decoding of mixtures to arrive at unambiguous groups. Further algorithms are provided for arriving at reasonable starting points for mixture estimation, through a modification to Bayesian model merging, as well as obtaining synchronous subgroups through the use of Viterbi decomposition.

We demonstrate biological relevance by the analysis of HeLa and Yeast time-course data sets for which we infer synchronous groups specific to cell cycle phases. With the use



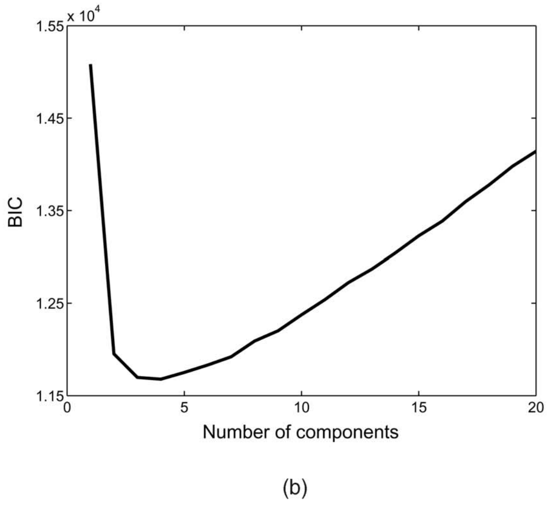


Fig. 11. We show BIC versus number of components for the data set (a) SIM1 and (b) Y5. The correct number of clusters are six and five, respectively.

of Gene Ontology information, we show the effectiveness of the mixture decoding. A comparison of different methods on simulated data, created under mild assumptions distinct from those implicit to other and our method, yielded favorable results. Our flexible framework, combined with an effective graphical user interface implemented in the GQL application, supports interactive, exploratory knowledge discovery, making full use of biological expert knowledge.

It should be noted that some of our contributions—mixture estimation in a partially supervised learning setting, subsequent inference of unambiguous groups—apply *in general* to any setting where clustering has so far been predominant, and that we would also expect favorable results in other application areas where the existence of partitions is not guaranteed.

Acknowledgments

The second and last author would like to acknowledge funding from the DAAD/CNPq (Brazil) and the BMBF through the Cologne University Bioinformatics Center (CUBIC), respectively. Thanks to Wasinee Rungsarityotin and Benjamin Georgi for work on the HMM library GHMM [57], the GHMM Python interface, and GQL [31], to Xue Li and Olof Persson for work on the Viterbi Decomposition and choosing initial models, and to Tim Beissbarth for helping with GOStat [48]. Thanks also to Terry Speed for helpful discussions. Thanks to two anonymous referees and Ben Rich for their valuable comments on the manuscript.

References

[1] Y. Yang, S. Dudoit, P. Luu, D. Lin, V. Peng, J. Ngai, and T. Speed, "Normalization for cDNA Microarray Data: A Robust Composite Method Addressing Single and Multiple Slide Systematic Variation," *Nucleic Acids Research,* vol. 30, no. 4, Feb. 2002.

[2] Z. Bar-Joseph, "Analyzing Time Series Gene Expression Data," *Bioinformatics,* vol. 20, no. 16, pp. 2493-2503, Nov. 2004.

[3] M. Eisen, P. Spellman, P. Brown, and D. Botstein, "Cluster Analysis and Display of Genome-Wide Expression Patterns," *Proc. Nat'l Academy of Science,* vol. 95, pp. 14, 863-868, 1998.

[4] A. Gasch, P. Spellman, C. Kao, O. Carmel-Harel, M. Eisen, G. Storz, D. Botstein, and P. Brown, "Genomic Expression Programs in the Response of Yeast Cells to Environmental Changes," *Molecular Biology of the Cell,* vol. 11, pp. 4241-4257, 2000.

[5] S. Tavazoie, J. Hughes, M. Campbell, R. Cho, and G. Church, "Systematic Determination of Genetic Network Architecture," *Nature Genetics,* vol. 22, pp. 281-285, 1999.

[6] S.A. Rifkin and J. Kim, "Geometry of Gene Expression Dynamics," *Bioinformatics,* vol. 18, no. 9, pp. 1176-1183, Sept. 2002.

[7] I.L. MacDonald and W. Zucchini, *Hidden Markov and Other Models for Discrete-Valued Time Series.* London: Chapman & Hall, 1997.

[8] Z. Bar-Joseph, G. Gerber, D.K. Gifford, and T.S. Jaakkola, "A New Approach to Analyzing Gene Expression Time Series Data," *Proc. Sixth Ann. Int'l Conf. Research in Comp. Molecular Biology,* 2002.

[9] M.F. Ramoni, P. Sebastiani, and I.S. Kohane, "Cluster Analysis of Gene Expression Dynamics," *Proc. Nat'l Academy of Science,* vol. 99, no. 14, pp. 9121-9126, July 2002.

[10] M.F. Ramoni, P. Sebastiani, and P.R. Cohen, "Bayesian Clustering by Dynamics," *Machine Learning,* vol. 47, no. 1, pp. 91-121, Apr. 2002.

[11] K.Y. Yeung, C. Fraley, A. Murua, A.E. Raftery, and W.L. Ruzzo, "Model-Based Clustering and Data Transformations for Gene Expression Data," *Bioinformatics,* vol. 17, no. 10, pp. 977-987, 2001.

[12] P.T. Spellman, G. Sherlock, M.Q. Zhang, V.R. Iyer, K. Anders, M.B. Eisen, P.O. Brown, D. Botstein, and B. Futcher, "Comprehensive Identification of Cell Cycle-Regulated Genes of the Yeast Saccharomyces Cerevisiae by Microarray Hybridization," *Molecular Biology of the Cell,* vol. 9, no. 12, pp. 3273-3297, Dec. 1998.

[13] A. Krogh, M. Brown, I.S. Mian, K. Sjolander, and D. Haussler, "Hidden Markov Models in Computational Biology. Applications to Protein Modeling," *J. Molecular Biology,* vol. 235, no. 5, pp. 1501-1531, Feb. 1994.

[14] P. Smyth, "Probabilistic Model-Based Clustering of Multivariate and Sequential Data," *Proc. Seventh Int'l Workshop AI and Statistics,* D. Heckerman and J. Whittaker, eds., 1999.

[15] B. Knab, "Erweiterungen von Hidden-Markov-Modellen zur Analyse ökonomischer Zeitreihen," PhD dissertation, Informatik, Universität zu Köln, 2000.

[16] S.G.I. Cadez and P. Smyth, "A General Probabilistic Framework for Clustering Individuals," *ACM SIGKDD 2000 Proc.,* 2000.

[17] B. Wichern, "Hidden-Markov-Modelle zur Analyse und Simulation von Finanzzeitreihen," PhD dissertation, Informatik, Universität zu Köln, 2001.

[18] B. Knab, A. Schliep, B. Steckemetz, and B. Wichern, "Model-Based Clustering with Hidden Markov Models and Its Application to Financial Time-Series Data," *Between Data Science and Applied Data Analysis,* M. Schader, W. Gaul, and M. Vichi, eds., Springer, pp. 561-569, 2003.

[19] V. Castelli and T.M. Cover, "On the Exponential Value of Labeled Samples," *Pattern Recognition Letters,* vol. 16, pp. 105-111, 1994.

[20] M. Seeger, "Learning with Labeled and Unlabeled Data," Inst. for Adaptive and Neural Computation, technical report, Univ. of Edinburgh, 2001.

[21] M. Szummer and T. Jaakkola, "Partially Labeled Classification with Markov Random Walks," *Neural Information Processing Systems (NIPS),* vol. 14, 2002.

[22] A. Blum and S. Chawla, "Learning from Labeled and Unlabeled Data Using Graph Mincuts," *Proc. Int'l Conf. Machine Learning,* 2001.

[23] M. Belkin, "Problems of Learning on Manifolds," PhD dissertation, Univ. of Chicago, 2003.

[24] F.G. Cozman, I. Cohen, and M.C. Cirelo, "Semi-Supervised Learning of Mixture Models," *Proc. 20th Int'l Conf. Machine Learning (ICML),* 2003.

[25] V. Vapnik, *The Nature of Statistical Learning Theory.* Wiley, 1998.

[26] S. Basu, M. Bilenko, and R.J. Mooney, "A Probabilistic Framework for Semi-Supervised Clustering," *Proc. 10th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining (KDD),* Aug. 2004.

[27] L.R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proc. IEEE,* vol. 77, no. 2, pp. 257-285, Feb. 1989.

[28] C. Burge and S. Karlin, "Prediction of Complete Gene Structures in Human Genomic DNA," *J. Molecular Biology,* vol. 268, no. 1, pp. 78-94, Apr. 1997.

[29] A. Schliep, A. Schönhuth, and C. Steinhoff, "Using Hidden Markov Models to Analyze Gene Expression Time Course Data," *Bioinformatics,* vol. 19, no. 1, pp. 255-263, July 2003.

[30] A. Schliep, C. Steinhoff, and A. Schönhuth, "Robust Inference of Groups in Gene Expression Time-Courses Using Mixtures of HMM," *Bioinformatics,* vol. 20, no. 1, pp. 283-289, July 2004.

[31] I.G. Costa, A. Schonhuth, and A. Schliep, "The Graphical Query Language: A Tool for Analysis of Gene Expression Time-Courses," *Bioinformatics,* vol. 21, no. 10, pp. 2544-2545, 2005.

[32] A. Dempster, N. Laird, and D. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *J. Royal Statistical Soc.: Series B,* vol. 39, pp. 1-38, 1977.

[33] A. Stolcke and S. Omohundro, "Hidden Markov Model Induction by Bayesian Model Merging," *Proc. Neural Information Processing Systems 5 (NIPS-5),* 1992.

[34] A. Schliep, "Learning Hidden Markov Model Topology," PhD dissertation, Center for Applied Computer Science, Univ. of Cologne, 2001.

[35] B.H. Juang and L.R. Rabiner, "A Probabilistic Distance Measure for Hidden Markov Models," *AT&T Technical J.,* vol. 64, no. 2, pp. 391-408, 1985.

[36] W. Pedrycz, "Fuzzy Sets in Pattern Recognition: Methodology and Methods," *Pattern Recognition,* vol. 23, nos. 1/2, pp. 121-146, 1990.

[37] G. McLachlan and K. Basford, *Mixture Models: Inference and Applications to Clustering.* New York, Basel: Marcel Dekker, Inc., 1988.

[38] G. McLachlan and D. Peel, *Finite Mixture Models,* Wiley Series in Probability and Statistics. New York: Wiley, 2000.

[39] C. Wu, "On the Convergence of the EM Algorithm," *Annals of Statistics,* pp. 95-103, 1983.

[40] R. Boyles, "On the Convergence of the EM Algorithm," *J. Royal Statistical Soc.: Series B,* pp. 47-50, 1983.

[41] J.A. Bilmes, "A Gentle Tutorial of the EM Algorithm and Its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models," Technical Report TR-97-021, Int'l Computer Science Inst., Berkeley, Calif., 1998.

[42] H. C, E. Loken, and J.L. Schafer, "Difficulties in Drawing Inferences with Finite-Mixture Models: A Simple Example with a Simple Solution," *Am. Statistician,* vol. 58, no. 2, pp. 152-158, 2004.

[43] K. Nigam, A.K. McCallum, S. Thrun, and T. Mitchell, "Text Classification from Labeled and Unlabeled Documents Using EM," *Machine Learning,* 1999.

[44] G. Schwarz, "Estimating the Dimension of a Model," *The Annals of Statistics,* vol. 6, pp. 461-464, 1978.

[45] T.G.O. Consortium, "Gene Ontology: Tool for the Unification of Biology," *Nature Genetics,* vol. 25, pp. 25-29, 2000.

[46] F. Sokal and R.R. Rohlf, *Biometry.* New York: W.H. Freeman and Company, 1995.

[47] A. Reiner, D. Yekutieli, and Y. Benjamini, "Identifying Differentially Expressed Genes Using False Discovery Rate Controlling Procedures," *Bioinformatics,* vol. 19, no. 3, pp. 368-375, 2003.

[48] T. Beissbarth and T.P. Speed, "GOstat: Find Statistically Overrepresented Gene Ontologies within a Group of Genes," *Bioinformatics,* vol. 20, no. 9, pp. 1464-1465, 2004.

[49] V.R. Iyer, M.B. Eisen, D.T. Ross, G. Schuler, T. Moore, J.C. Lee, J.M. Trent, L.M. Staudt, J.R. Hudson, M.S. Boguski, D. Lashkari, D. Shalon, D. Botstein, and P.O. Brown, "The Transcriptional Program in the Response of Human Fibroblasts to Serum," *Science,* vol. 283, no. 5398, pp. 83-87, Jan. 1999.

[50] R. Cho, M. Campbell, E. Winzeler, L. Steinmetz, A. Conway, L. Wodicka, T. Wolfsberg, A. Gabrielian, D. Landsman, J. Lockhart, and W. Davis, "A Genome-Wide Transcriptional Analysis of the Mitotic Cell Cycle," *Molecular Cell,* vol. 2, pp. 65-73, 1998.

[51] S. Chu, J. DeRisi, M. Eisen, J. Mulholland, D. Botstein, P.O. Brown, and I. Herskowitz, "The Transcriptional Program of Sporulation in Budding Yeast," *Science,* vol. 282, no. 5389, pp. 699-705, 1998.

[52] M.L. Whitfield, G. Sherlock, A.J. Saldanha, J.I. Murray, C.A. Ball, K.E. Alexander, J.C. Matese, C.M. Perou, M.M. Hurt, P.O. Brown, and D. Botstein, "Identification of Genes Periodically Expressed in the Human Cell Cycle and Their Expression in Tumors," *Molecular Biology of the Cell,* vol. 13, no. 6, pp. 1977-2000, June 2002.

[53] A.K. Jain and R.C. Dubes, *Algorithms for Clustering Data.* Prentice Hall Int'l, 1998.

[54] C.M. C. Milligan G.W., "A Study of the Comparability of External Criteria for Hierarchical Cluster Analysis," *Multivariate Behavior Research,* vol. 21, pp. 441-458, 1986.

[55] B. Kraus, M. Pohlschmidt, A.L. Leung, G.G. Germino, A. Snarey, M.C. Schneider, S.T. Reeders, and A.M. Frischauf, "A Novel Cyclin Gene (CCNF) in the Region of the Polycystic Kidney Disease Gene (PKD1)," *Genomics,* vol. 24, no. 1, pp. 27-33, Nov. 1994.

[56] M. de Hoon, S. Imoto, J. Nolan, and S. Miyano, "Open Source Clustering Software," *Bioinformatics,* vol. 20, no. 9, pp. 1453-1454, 2004.

[57] "The General Hidden Markov Model Library (GHMM)," http://ghmm.org, 2003.

Alexander Schliep received the PhD degree in computer science from the Center for Applied Computer Science at the University of Cologne, Germany, in 2001, working in collaboration with the Theoretical Biology and Biophysics Group (T-10) at Los Alamos National Laboratory. Since 2002, he has been group leader of the bioinformatics algorithms group in the Department for Computational Biology at the Max Planck Institute for Molecular Genetics in Berlin. The research interests pursued in his group include data mining, statistical models, and algorithms for analyzing complex, heterogeneous data from molecular biology.

Ivan G. Costa received the MSc and BSc degrees in computer science at the Universidade Federal de Pernambuco, Brazil. Since 2004, he has been working toward the PhD degree in computer science in the Department for Computational Biology at the Max Planck Institute for Molecular Genetics in Berlin. His research interests are in the area of pattern recognition, statistical learning, and cluster validation, as well as applications of these in transcriptomics.

Christine Steinhoff studied mathematics and genetics at Heinrich Heine University in Düsseldorf and at Oxford University. She received the PhD degree in molecular biology in 2001 working on gene therapeutical models. Afterward, she joined the bioinformatics group at the Max Planck Institute for Molecular Genetics in Berlin. Her research interests include different statistical aspects of data analysis of microarrays and epigenetic gene regulation models.

Alexander Schönhuth joined the Bioinformatics Group as a PhD student in the Center for Applied Computer Science at the University of Cologne, Germany, at the end of 2001. He is a scientific assistant in the Cologne University Bioinformatics Center holding Computer Science lectures and guiding project work. Educated in pure Mathematics, he has specialized in Markovian models and their extensions in statistical learning theory. Biological applications of his work are sequence analyses of proteins as well as mining gene expression time-courses.