Chapter 4

Analysis of Gene Expression Time Courses

The analysis of gene expression over the course of time is an important step to understanding function and regulatory roles of genes [11]. For example, during the cell cycle, we can find groups of genes that are only over-expressed in a particular phase. From this information, one can try to infer the gene function by relating it to genes with known function and similar expression profiles in a "guilty by association" approach [71]. Another alternative is to explore the promoter sequence of a group of co-expressed genes and search for common patterns of transcription factor binding sites in the upstream region of those genes [201]. Furthermore, by searching for genes (or groups) with similar expression profiles patterns, but with distinct time of change in gene expression levels, it is possible to explore regulatory roles [78]. For instance, earlier "activated genes" could regulate genes with similar expression patterns with a later "activation" time. A first step towards all these analyses is to find co-expressed genes, or groups of genes that display a similar expression profile over the course of time.

In this context, initial work was based on clustering methods that assume independence between expression values of distinct time points, for example, hierarchical clustering [71, 82], k-means clustering [212] and singular value decomposition [6]. However, in a temporal setting, the expression value in a time point is dependent on values of preceding time points. In contrast to these earlier studies, temporal models were applied to gene expression time courses, such as cubic splines [12, 138] and autoregressive curves [172]. These methods are often robust to noise found in time course experiments, such as noise in a single time point or expression profiles showing a slower rate of expression. Furthermore, these methods can make use of information relevant to time courses such as sampling time and periodicity.

As one of the main contribution of this thesis, we propose in this chapter a hidden Markov model (HMM) with a linear topology that is suitable for modeling time-dependent sequences such as a time courses. These models represent co-regulated genes with a similar prototypical behavior, or the same sequence of expression level changes, in an asynchronous manner. By asynchronous, we mean that the HMM captures time courses with the same events of expression changes at possibly distinct time points. Nevertheless, syn-

chronous groups and their time of expression changes can be latter inferred from a model by the analysis of the most probable state path of a time course in the HMM. Therefore, it is possible to build a partial order on synchronous groups of genes all displaying the same prototypical expression pattern but with distinct time of expression changes.

Two main applications of linear HMMs are presented here. The first is an iterative graphical tool that allows an user to query a set of gene expression time courses for those displaying a specific prototypical behavior. This tool allows the user to build a linear HMM interactively and to explore a given gene expression data set for prototypical patterns of interest. A second, and more intricate application, is to estimate a mixture of linear HMMs to find groups of genes with the same prototypical expression patterns within a given data set.

This chapter is organized as follows. First, we describe related work in Section 4.1. Then, we give a brief definition of HMMs (Section 4.2) and the specific HMM topology employed here. In Section 4.3, we describe the application for querying gene expression data with a linear HMM, while in Section 4.4 we introduce a method for finding groups of genes with a mixture of HMMs. In Section 4.5, we present the evaluation of our method with two gene expression data sets. We present final remarks and future directions in Chapter 7.

4.1 Related Work

There are several computational tasks related to the analysis of gene expression time courses. These can be divided in three levels: (1) pre-processing methods, which are responsible for tasks such as microarray data normalization; (2) exploratory methods, which search for genes differentially expressed or for groups of co-regulated genes; and (3) network-based methods, which try to reconstruct regulatory (or metabolic) networks from genes (or gene products). For methods in (1) and (2), it is a common strategy to take the temporal nature of time courses into account. Next, we will give an overview of the most relevant methods in the exploratory level, which is the category our method belongs to. More complete reviews of methods in all these analyses levels can be found in [8, 11], and methods for normalization and differential expression in [29, 203].

There are few approaches concerning the detection of differentially expressed genes from time course gene expression. For example, [206] extended the significant analysis method in [219] to take temporal dependencies into account. In [208] an Empirical Bayes approach was presented to detect differentially expressed genes when replications of the time course measurements are present.

One interesting variation of the problem described in the previous paragraph is the detection of genes displaying different temporal expression patterns in time courses measured under distinct conditions, e.g., experiments with two (or more) time courses, each one measuring a particular cell applied to a distinct treatment or an environmental condition. Genes differentially expressed in one or more time courses should be biologically relevant to the treatment or condition analyzed. In [13], for instance, data from time courses from yeast

cell cycle of a wild type and after FKh1 or FKH2 knockout were studied. There, B-splines were used to model the temporal patterns. Analysis of promoter regions confirmed that the detected differentially expressed genes were related to the knockout factors. A similar problem was approached in [235], where gene expression time courses of mice with distinct oxidative stress and age were investigated. Their method was based on a HMM with linear topology and Gamma distributions as state emissions. Using an Empirical Bayes approach [36], the authors could obtain p-values concerning the significance of these differences.

One particular type of time course data, which has received great attention in the literature, is gene expression time courses of the cell cycle, which is the process through an eukariotic cell duplicates into two genetically identical cells. The cell cycle consists of four phases: G1 phase, where the cell starts to grow and prepare for DNA replication; S phase, where DNA replication occurs; G2 phase, where microtubules are produced; and, finally, M phase, where nuclear and cytoplasmic division occurs [4]. For these experiments, a particular population of cells is arrested at a cell cycle phase (usually G1). After the release of the cells to start the cell cycle, the expression is measured over the course of two to three cell cycles. Examples of such data sets are found in [42, 201] for yeast and in [226] for human HeLa cells. In such data sets, some time courses have a periodic behavior usually displaying a cosine type of gene expression pattern, with an over-expression peak at one particular phase at each cell cycle (see Figure 4.11 for an example of such profiles). One particular effect of the arresting protocol is that the individual cells will have distinct cell cycle periods. Thus they will desynchronize, and the periodic signal deteriorates with time. Also, such data sets have usually more than 30 time points, a number far larger than most other time course experiments [73].

Such cell cycle based experiments pose a number of interesting methodological questions. In [1], a method for aligning pairs of time courses with a dynamic programming algorithm was proposed. Such a method finds pairs of genes, which have a similar expression pattern, but distinct phases. Pairs displaying such time-lags (or shift in expression patterns) are potential candidates for regulatory relationships. A more extensive study was performed in [78], where methods for shift and phase detections were proposed, so that distinct cell cycle experiments could be integrated in one.

One typical application in cell cycle data sets is the detection of transcripts displaying periodic behavior, as well as at which time point these periodic transcripts peak. See [58], for a good review of methods and comparison on popular data sets. Interestingly, [58] found that a simple method based on Fourrier analysis [201] was significantly better than all more sophisticated computational approaches. Also, a recent work has shown that the choice of the background model for performing the statistical tests has a great impact on the detection of periodic expressed genes [81].

In [71], the problem of finding functional modules of genes with the use of clustering methods was first proposed. This study is restricted to the application of simple methods, such as k-means and hierarchical clustering with classical distance measures such as Euclidean

distance or the Pearson correlation coefficient. Its main methodological contribution was the proposal of a heat-map style plot for displaying groups of co-regulated genes, also known as red and green plots (see Figure 5.5 for an example of such plots). More recently, [6] proposed the use of singular value decomposition and [232] the use of mixture of multivariate Gaussians. In the latter, a wide comparison on the effect of data normalization strategies and the choice of the covariance matrix on the results was performed in a cell cycle data set. It was shown that full covariances matrices were prone to over-fitting, and that standardization of the time courses improved the results. One alternative to the use of classical distance measures is the use of similarity measures based on mutual information. In contrast to Pearson correlation and Euclidean distance, Mutual information is able to capture non-linear correlations (or dependencies) [33, 204]. Nevertheless, these methods are based on expression data sets with large number of samples, i.e., more than > 100 biological observations, that is rarely the case in gene expression studies.

None of the previous reviewed methodologies take the temporal nature of time courses into consideration for finding co-regulated genes. In particular, classical clustering methods, such as k-means and hierarchical clustering rely on distance functions between expression profiles, such as correlation or Euclidean distance, which neglect the temporal nature of time-courses. This shortcoming was first approached in [172], where auto-correlation models based on a low-order linear regression was proposed. This clustering method works in an agglomerate Bayesian fashion. It relies on several heuristics for the choice of parameters and finding the number of clusters. Also, another shortcoming of this method is the modeling of only low order dependencies, which will fail in modeling correlations in cell cycle data sets, where periodicity occurs after 8 or more time points. Simultaneously [12] and [138] proposed model-based clustering approaches using cubic splines [12] and B-splines [138]. Both methods require the specification of several parameters for the splines and take advantage of time courses with a large number of time points. The use of a model called hidden phase model (HPM) as a prior on a mixture of multivariate Gaussians was proposed in [30]. The HPM model, which can be seen as an extension to an HMM, allows the user to include preference towards component models with a particular type of temporal pattern, e.g., cyclic patterns, increase in expression, or decrease in expression. The method displayed good performance in recovering clusters of periodically expressed genes, even when a low number of time courses were present.

The method proposed in this chapter, mixture of linear HMMs, has a similar motivation as the methods described in the previous paragraph [12, 138, 172]. As in those methods, linear HMMs makes use of temporal dependencies for modeling groups of co-expressed genes. The main distinction of our method to those is the capability of modeling groups of genes with similar expression patterns, but with asynchronous changes in expression. As each of these approaches are based on distinct assumptions for modeling time dependencies, none of them is likely to be the overall best choice. For comparing these methods with our approach, we perform an empirical evaluation using data from yeast cell cycle data set described in Section 4.5.1.

In [234], it was investigated how repeated microarray measurements could be integrated in the cluster analysis. The rationale behind their approach is that measurements with low replicate variability should be more reliable than measurements with high variability. Consequently, they have a higher weight on the clustering procedures. They applied this idea on several model-based clustering methods. The study found that an infinite mixture of multivariate Gaussians obtained the most favorable results. However, the work did not explore any method modeling the temporal dependencies. Even though our proposal do not explore replicate estimates, an extension of our method using the proposal of [234] is straightforward.

Recently, attention has been given to the fact that time courses data sets have usually few time points [73, 150]. Indeed, 80% of the data sets in Gene Expression Omnibus [69] have less the 8 time points. This is of crucial importance, as most of the model-based clustering methods described before [12, 138, 172] will suffer from over-fitting with such data sets. In [150], a fuzzy clustering method for short and unevenly sampled time courses was proposed. This method takes into account first-order dependencies and the sampling of time points. They showed for data set with seven time points that the method was superior to k-means and hierarchical clustering. In [73], the authors proposed a method that performs a greedy search over the set of possible expression patterns. It finds the patterns that are significantly distinct from the others. The authors showed that the method had favorable results in relation to k-means and CAGED [172] for a time course data set with 5 time points. Note that even thought this also poses a problem the mixture of HMMs, which also take advantages of larger time courses, this point could be tackled in our approach with the use of structural learning techniques favoring HMMs with fewer stages.

An interesting problem is the integration of additional biological data in the detection of groups of co-regulated genes. In [101], authors explored the joint analysis of gene expression and sequences from promoter regions. The main idea is to inspect if co-regulated genes also shared similar transcription factor binding sites. They defined a method based on the EM and Gibbs sampling for performing a clustering with both expression and sequence data. The same problem was approached with an EM method in [194] with the use of discriminative position weights matrices as models for hits of transcription factor binding sites. However, none of these approaches used models taking temporal dependencies into account. One exception is the work of [231], which combined location analysis data (also known as Chip-on-chip [128]) with time courses from cell cycle. They could find groups of co-regulated genes displaying time lagged expression pattern in relation to the expression profile of transcription factors. Data from location analysis confirmed regulatory roles between some of these transcription factors and groups of co-expressed genes. In [74], the authors applied an input-output HMM to combine time course gene expression with location analysis data in order to detect groups of genes, which are targets of a given transcription factor and have a similar co-expression profile. They analyzed short time courses after treatment of yeast to stress conditions, and could detect novel putative regulatory roles.

Recently, there has been a great deal of data sets with multiple (usually short) time courses measured over distinct gene knockouts [236], treatments [174], patients [225], or environmental conditions [82]. Such data present new methodological challenges not addressed before. In [113], time courses from multiple sclerosis patients after particular treatments were analyzed. Their Bayesian framework could detect gene responses, which were specific to either the treatment type or to specific responses of a particular patient. In [174], time courses of Arabidopsis after several distinct treatments were analyzed. The work aimed to find genes that display a treatment specific time-lag in their expression profiles in relation to the expression profile of known transcription factors. The groups of time-courses with time-lag were found with the use of a covariance index and an heuristic based on the Gap statistic. In [197], a similar problem was approached with the use of a graphical model. They used a set of known transcription factors and gene target relations to estimate parameters and time lags. Then, their model was used to predict unknown regulatory relationships. One main difficulty of this problem is the fact that the time lag is dependent of the biological condition and transcription factor, which requires the estimation of a large number of free parameters. However, both [174, 197] showed that their methods, by using multiple time courses, were superior is detecting regulatory roles of genes from expression time-lag than methods based on single time course data as [1, 78].

4.2 Hidden Markov Models

A hidden Markov model (HMM) is a probabilistic model composed of a Markov chain with M discrete states and emission probability density functions (pdf) assigned to each state. At a given time point, a HMM is at a particular unknown state and it emits a symbol in accordance to the density function assigned to that state. More formally, given a continuous random variable $X = (X_1, ..., X_t, ..., X_L)$ representing the emitted symbols, and a discrete hidden variable $Q = (Q_1, ..., Q_t, ..., Q_L)$. For an given observation $x = (x_1, ..., x_t, ..., x_L)$ from X, we have a corresponding hidden sequence path $q = (q_1, ..., q_t, ..., q_L)$, where $q_t \in \{1, ..., M\}$ represents the state emitting x_t. A HMM allows a computational efficient approximation of the joint densities $p(x, q)$ for observations x and q. There are two main independence assumptions regarding HMMs: (1) the probability to reach a state t depends only on the previous state $(t-1)$ [1]

$$p(q_t|q_1, ..., q_{t-1}) = p(q_t|q_{t-1}), \tag{4.1}$$

and (2) the density function of emitting x_t depends only on the current state t

$$p(x_t|q_1, ..., q_t) = p(x_t|q_t). \tag{4.2}$$

We can represent the probabilities in Eq. 4.1 by a transition matrix $A = \{a_{uv}\}$ for $1 \leq u \leq$

[1] We only consider here, HMMs with first-order dependencies.

M and $1 \leq v \leq M$, where a_{uv} is equal to the probability of going from state u to state v, i.e., $p(q_t = v|q_{t-1} = u)$, given that $\sum_{v=1}^{M} a_{uv} = 1$ and $a_{uv} \geq 0$ for $1 \leq u \leq M$. The initial state probabilities $p(q_1 = u) = \pi_u$ are represented by a vector $\pi = (\pi_1, ..., \pi_u, ..., \pi_M)$. In our problem, which deals with gene expression, the emission variables are continuous, and univariate Gaussian densities are used as emission function on the states

$$p(x_t|q_t = u) = p_u(x_t|\mu_u, \sigma_u^2) = \frac{1}{\sqrt{2\pi\sigma_u^2}} \exp - \frac{(x_t - \mu_u)^2}{2\sigma_u^2}, \tag{4.3}$$

where $p_u(x_t|\mu_u, \sigma_u^2)$ is the probability density function (pdf) associated with the uth state, and μ_u, σ_u^2 are the pdf parameters. Hence, a HMM with M states is parameterized by the vector $\theta = (A, ((\mu_1, ..., \mu_M), (\sigma_1^2, ..., \sigma_M^2)), \pi)$.

Given an observation x and the corresponding sequence of visited states, the joint distribution can be defined as follows

$$\begin{aligned} p(x, q|\theta) &= p(x|q, \theta)p(q|\theta) && (4.4)\\ &= p(q_1)p(x_1|q_1) \prod_{t=2}^{L} p(q_t|q_{t-1})p(x_t|q_t) && (4.5)\\ &= \pi_{q_1} f_{q_1}(x_1|\mu_{q_1}, \sigma_{q_1}^2) \prod_{t=2}^{L} a_{q_{t-1},q_t} f_{q_t}(x_t|\mu_{q_t}, \sigma_{q_t}^2) && (4.6) \end{aligned}$$

For a given HMM defined by the parameters θ, one first natural question, following [169], is how to compute the likelihood of an observation x, or

$$p(x|\theta) = \sum_{q \in \mathcal{Q}} p(x, q|\theta), \tag{4.7}$$

where $\mathcal{Q}$ is the set of all possible state sequences q. A brute force calculation of the previous equation requires the infeasible evaluation of M^L sequences. Nevertheless, a technique based on dynamic programming, called forward-backward algorithm allows us to compute Eq. 4.7 in $O(ML)$ time [20].

The second problem is the maximum likelihood estimation of a HMM from a data set $\mathbf{X}$ with N observations, where x_i is the ith observation from $\mathbf{X}$, that is, finding

$$\theta^* = \arg\max_{\theta} \sum_{i=1}^{N} p(x_i|\theta). \tag{4.8}$$

This problem can be solved with the Baum-Welch algorithm [21], which is a specific application of the EM algorithm for HMMs.

The last task is the decoding problem, or for a given observation x and a HMM with

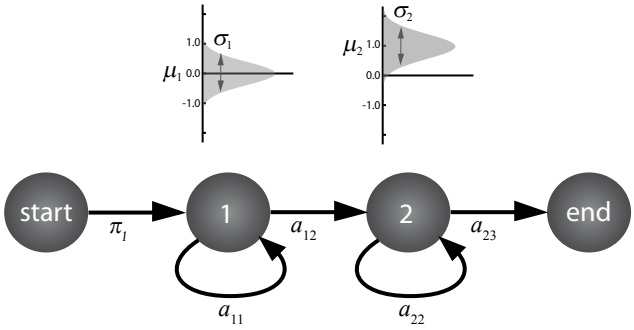


Figure 4.1: *Example of a linear HMM with two emitting states (states 1 and 2), an initial state and an end state. Given the linear topology, all sequences will always visit, respectively, the start state, 1, 2 and end state. The state 1 models expression values near zero, while the state 2 models expression values around one. Thus, this lHMM models time courses up-regulated at some time point. The values of the self-transition parameters a_{11} and a_{22} will define the expected duration that a state will be visited. For example, if we set $a_{11} > a_{22}$, the model will give a higher likelihood for expression profiles with late up-regulation patterns.*

parameters θ, find the sequence of visited states q that has maximum likelihood,

$$q^* = \arg\max_q p(x, q|\theta). \tag{4.9}$$

This problem can be solved with the Viterbi algorithm [222]. For more details, proofs and other extensions we refer to [121, 169].

4.2.1 Linear HMM and Time Courses

We use a particular linear topology for modeling gene expression time courses. In this topology, a given state only has a self-transition and a transition to the next state. We include in the beginning (and end) of each chain a special start state (and end state). Given that the time courses are normalized by taking the logarithm in relation to a control experiment (an usual procedure in gene expression time course analysis [71, 201]), we interpret that an expression value near zero means expression close to background level, above zero over-expression (or up-regulation) and below zero under-expression (or down-regulation). For example, a linear HMM—lHMM for short—with two emitting states: the first ones with a mean emission of zero and the second one with a mean emission of one, we model time courses displaying an up-regulation prototypical behavior (Figure 4.1).

More formally, the random variable $X = (X_1, ..., X_t, ..., X_L, X_{L+1})$ represents the gene expression time courses, where $x = (x_1, ..., x_t, ..., x_L, x_{L+1})$ is an observed time course, x_t is the expression value at time-point t and x_{L+1} is set to the special ending symbol "$\ltimes$". As described in the Section 4.2, a HMM is parameterized by a transition matrix

A, emission parameters $(\mu_1, ..., \mu_M), (\sigma_1^2, ..., \sigma_M^2)$ and initial probability vector π. As we restrict the topology to a linear chain of states, we can adopt a simple notation for the linear HMMs. For a given state u, we only need to define the self-transition probability a_{uu}, as $a_{u(u+1)} = 1 - a_{uu}$ and all other transitions from state u are set to zero. A more intuitive representation is the expected duration length of a state. For a single state, the length distribution respects a geometrical distribution [67], and has an expected length of

$$d_u = \frac{1}{1 - a_{uu}}. \tag{4.10}$$

This parameter represents how many time steps the observation sequence is expected to spend in state u. Accordingly, we can obtain the transition probability a_{uu} from a given duration,

$$a_{uu} = 1 - \frac{1}{d_u}. \tag{4.11}$$

We also constrain the duration parameters so that

$$\sum_{u=1}^{L} d_u = L, \tag{4.12}$$

for assuring that the expected length of reaching the end state matches the time course length L.

We include an end state $M + 1$ with a self-transition set to one and an emission density fixed to a special ending symbol. The initial probability vector π can be ignored, since we will have $\pi_1 = 1$ and $\pi_u = 0$ for $2 \leq u \leq M$. Given that all observations x have an end symbol at x_{L+1} and $M \leq L + 1$, all sequence paths not visiting each state at least once have zero likelihood. Consequently, such sequence paths are ignored in the parameter estimation performed by the Baum-Welch algorithm. Furthermore, we have the parameters μ_u and σ_u^2 for the emission function. Finally, a given state is parameterized by the triple (d_u, μ_u, σ_u^2), and a lHMM by $\theta^l = ((d_1, \mu_1, \sigma_1^2), ..., (d_M, \mu_M, \sigma_M^2))$.

Such linear HMMs are able to model time courses displaying several prototypical behaviors. For example, in Figures 4.1 and 4.2, we depict a lHMM that models profiles with distinct prototypical expression behaviors (up-regulation, up and down-regulation, and so on). One way to interpret the temporal behavior modeled by an lHMM is the following: each state represents a particular level of expression specified by μ_u, where a certain level of error (encoded by σ_u^2) is allowed[2], and the time course has an expected length of d_u of staying in this particular expression level. This can be interpreted as a "bounding box" specifying the expected expression of time courses, as depicted in Figure 4.3. Even though a lHMM models the expected duration of visiting states, it allows some flexibility concern-

[2]In a normal probability density function the interval $[\mu - \sigma, \mu + \sigma]$ defines the region around the mean containing 68% of the density. In the figures depicting the lHMMs, the lengths of the σ are only used for illustration purposes and do not strictly define the exact proportion of the densities.

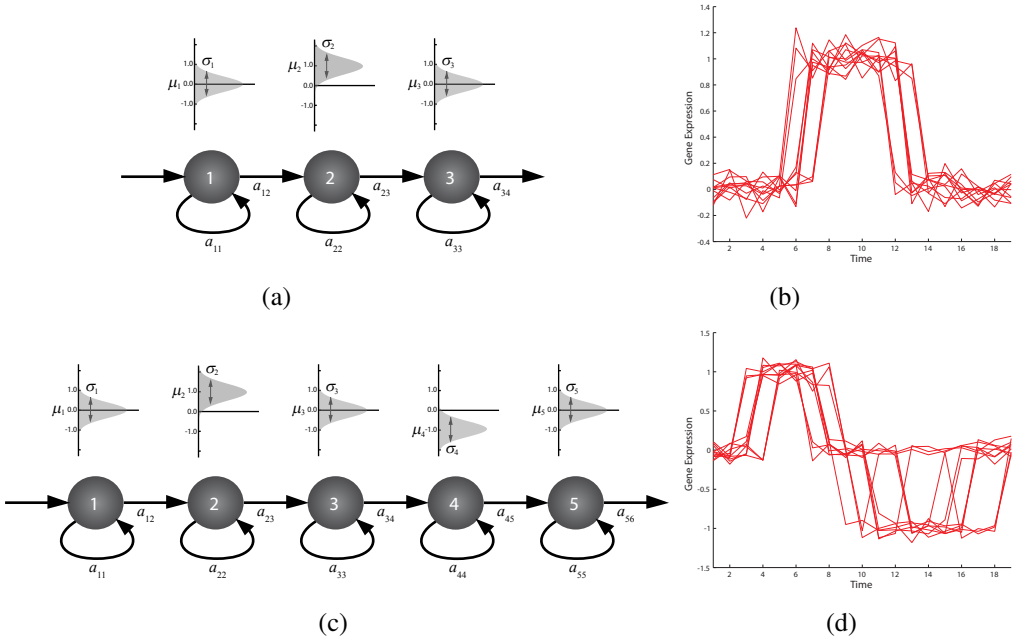


Figure 4.2: *Example of linear HMMs modeling up-regulation (a) and up- and down-regulation patterns (c). On (b) and (d), we display time courses with 19 time points observations from these linear HMMs. The duration of states are all set so that time courses will visit each state for the same number of time points. For simplicity, we do not depict start and end states from models.*

ing the time of changes in expression levels (or transitions between distinct states). This characteristic makes it possible to model asynchronous time courses showing a similar prototypical behavior. An inspection of the Viterbi path q for a given model θ^l and time course x (Eq. 4.9) indicates the most probable sequence of transitions and can be used to recover synchronous groups of time courses. For example, in Figure 4.4, the Viterbi paths indicate the existence of three synchronous groups of up-regulated genes: up-regulation event from the time point 9 to 10 (green), from time point 10 to 11 (red) and from time point 11 to 12 (blue).

Periodic Gene Expression Time Courses. We can extend the linear HMM topology to model some special applications in gene expression time course analysis. For example, in periodic time courses, expression is measured during two or three cell cycles, and interesting genes should show a periodic or cyclic behavior [42, 201]. By including a transition probability from state M to state 1 and adjusting all d_u to match the expected cycle length, we obtain a HMM that models periodicity.

Formally, we need to specify the expected cycle length R and the number of cycles measured S. Both values are usually known from the experimental setting used in the gene expression data acquisition. The duration of states should be chosen such that $\sum_{u=1}^{M} d_u = R$. For the last emitting state L, a_{LL} is defined as usual (Eq. 4.11), and the other transitions

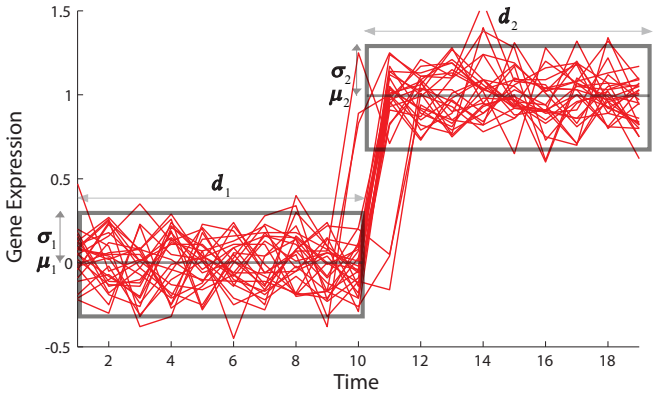


Figure 4.3: *Example of up-regulated time courses with 19 time points sampled from the linear HMM depicted in Figure 4.1. For that topology, we set the parameters of state 1 to* $(d_1, \mu_1, \sigma_1^2) = (10, 0.0, 0.1)$ *and of state 2 to* $(d_2, \mu_2, \sigma_2^2) = (9.0, 1.0, 0.1)$. *The duration value* d_1*=10 corresponds to setting* $a_{11} = 0.9$ *and* $a_{12} = 0.1$. *The gray boxes depict the overall expression pattern modeled by each state. For example, in state 1,* $\mu_1 = 0$ *defines the box location in the* y*-axis (or the mean expression value),* σ_1 *the width of the box in the* y*-axis (or the allowed variation around the mean expression value) and the parameter* d_1 *the length of the box in the* x*-axis (or the duration a time course will stay at a particular expression value). Note that the boxes define only the expected expression behaviors; time courses violating these bounds are allowed as well as long as the overall expression pattern is matched. Also, asynchronicity in the time courses is allowed, as we have time courses where the up-regulation event occurred a few points before or after the expected transition time point (t=10).*

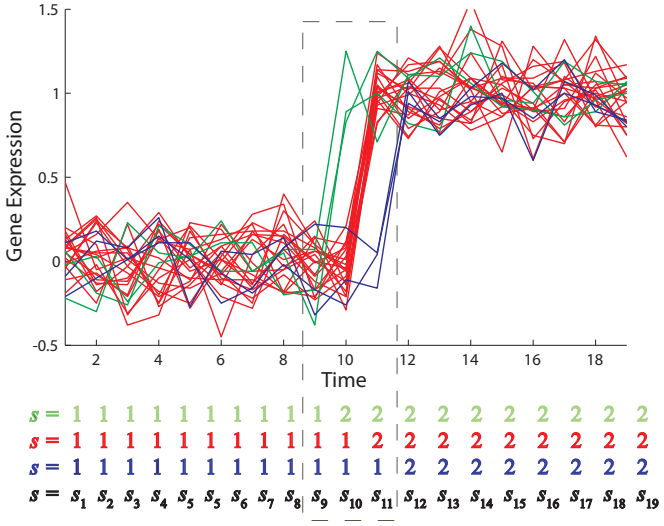


Figure 4.4: *Example of the Viterbi paths of the time courses from Figure 4.3 given the model depicted in Figure 4.1. These time courses have one of the three Viterbi paths: time courses in green have a transition from state 1 to state 2 at time point s_9 to s_{10}, time courses in red at s_{10} to s_{11} and time courses in blue from s_{11} to s_{12}. The Viterbi path sequences are depicted below the graph with the color of the corresponding group of time courses.*

are specified as follows

$$a_{L1} = \frac{(1 - a_{LL})(S - 1)}{S} \tag{4.13}$$

and

$$a_{L(L+1)} = \frac{1 - a_{LL}}{S}. \tag{4.14}$$

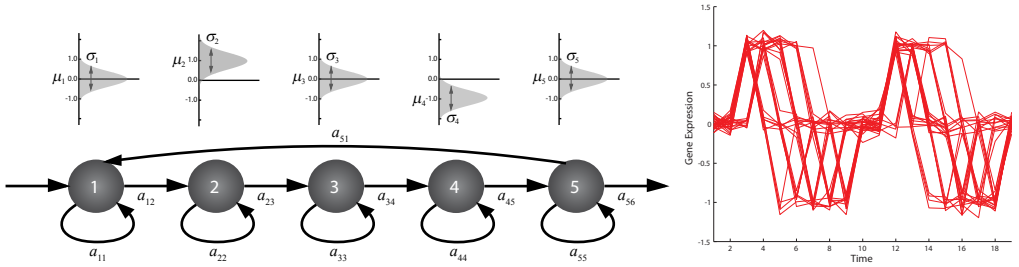
Support of Missing Data. Another useful extension, in the context of gene expression, is the support of missing data. Formally, we define a special symbol (Nan) and extend the space of variable X to $\mathbb{R} \cup \{\text{Nan}\}$. We then redefine the emission density from Eq. 4.3 as follows

$$p(x_t|q_t = u) = (1 - \phi) * f_u(x_t|\mu_u, \sigma_u^2) + \phi * 1(x_t = \text{Nan}), \tag{4.15}$$

where ϕ represents the proportion of missing observations. For given data $\mathbf{X}$, we can derive ϕ by measuring the percentage of missing symbols in $\mathbf{X}$

$$\hat{\phi} = \frac{\#\{x_{iu} = \text{Nan}|x_{iu}, 1 \leq i \leq N, 1 \leq u \leq L\}}{L \cdot N}. \tag{4.16}$$

Linear HMMs and Multivariate Gaussians. There is a close relation between the linear HMMs presented in this chapter and the multivariate Gaussians discussed in Section 2.3.3. In a lHMM with one state per time-point, which implies $M = L$ and $a_{uu} = 0$ for all



(a) (b)

Figure 4.5: *Example of the extension of the linear HMM defined in Figure 4.2 (a) for modeling periodicity (a). The model requires simply the addition of transition a_{51} and re-parameterization of all transition values. We display time courses with $L = 19$ sampled from this model (b).*

$1 \leq u \leq L$, the linear model is equivalent to a multivariate Gaussian with mean vector $(\mu_1, \ldots, \mu_L)$ and covariance matrix $diag(\sigma_1, \ldots, \sigma_L)$ (see Section 2.3.3). Typically, the number of states of a lHMM respects $M < L$ and there is no simple analytical way to parameterize a multivariate Gaussian from the parameters of a lHMM in this case. The reason for this is that the probability of visiting a state u is dependent on each time course, which would require the computation of posterior probabilities of being at a given state $(p(s_l = u|x)$ [67]).

One way to evaluate the type of covariance structures modeled by the linear HMMs is to sample data from a linear HMM, and estimate the empirical covariance matrix from it. In order to do so, we sampled 1,000 sequences with length $L = 19$ from the models depicted in Figures 4.1, 4.2 and 4.5, and estimated the covariance matrices as defined in Eq.2.28. We also sampled data from a linear HMM with 19 states, where $\mu_u = 0$ for $1 \leq u \leq 10$, $\mu_u = 1$ for $11 \leq u \leq 19$, $d_u = 1$ and $\sigma_u = 0.01$ for $1 \leq u \leq 19$. This lHMM, which measures time courses up-regulated from time point 10 to 11, corresponds to the case with $L = M$ and it is equivalent to a multivariate Gaussian with diagonal covariance matrix. The covariance matrices of four lHMMs are displayed in Figure 4.6. As expected, the covariance matrix derived from the lHMM with $L = M$ has values near zero (dark blue) in all off-diagonal entries. In other words, there is no dependence between time points. For the lHMM modeling up-regulation—Figure 4.2 (a)—the covariance matrix indicates correlations between consecutive time points (entries near the diagonal). The covariance matrix from the lHMM modeling up- and down-regulation—Figure 4.2(c)—has a more intricate correlation pattern, displaying correlation between consecutive time points, but also a negative correlation with time points 7 steps apart. This is a consequence of the down-regulation event, which happens later in these time courses. In the cyclic lHMM, the block of the matrix σ_{uv} for $1 \leq u \leq 9$ and $1 \leq v \leq 9$ (also σ_{uv} for $11 \leq u \leq 19$ and $11 \leq v \leq 19$) have a courser but similar covariance structure to the lHMM modeling up- and down-regulation (Figure 4.2 (c)), and all other entries have values similar to zero.

If we inspect only time courses following similar Viterbi paths, we get a further under-

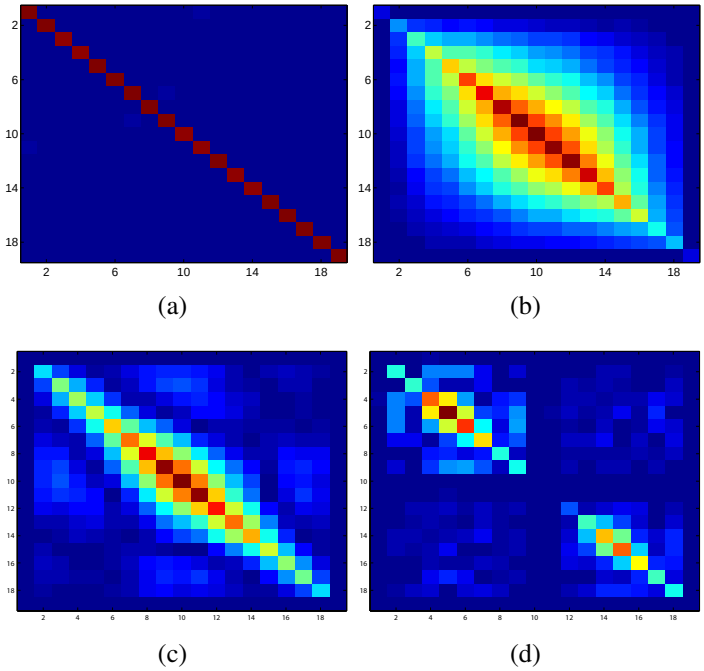
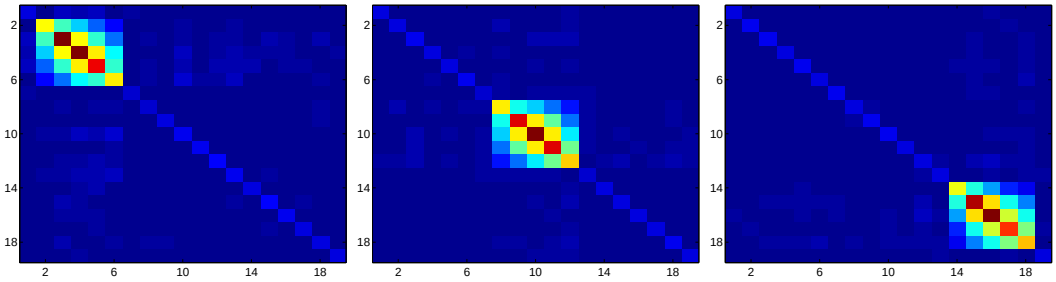


Figure 4.6: *We depict heat map plots of the absolute value of the empirical covariance of data sampled from a 19 state lHMM (a), from the up-regulation lHMM depicted in Figure 4.1 (b), up- and down-regulation lHMM depicted in Figure 4.2 bottom (c) and the cyclic lHMM depicted in Figure 4.5 (d). Dark blue entries indicates zero values, while red indicates positive (or negative) values.*

standing of the covariance structure modeled by the lHMM. For example, if we look at the simulated data from the lHMM depicted in Figure 4.1, and divide it into three groups: (1) time courses with the up-regulation event before time point 7; (2) time courses with up-regulation between 7 and 12; and (3) time courses up-regulated after time point 12. The covariance matrices of these groups are depicted in Figure 4.7. The resulting covariance matrices are decompositions of the original matrix—Figure 4.6 (b), where the covariances between consecutive time points are restricted now to earlier (Figure 4.7 (a)), middle (Figure 4.7 (b)) or late (Figure 4.7 (c)) time points.

The linear HMMs allow modeling of dependencies between subsequent time points. Moreover, depending on the topology, longer term dependencies are also captured, e.g., up- and down-regulation lHMM (Figure 4.2 (c)). One very important characteristic is the few number of free parameters necessary for the linear HMMs. A lHMM requires the specification of $(3 \times M - 1)$ free parameters. In contrast, a multivariate Gaussian with diagonal matrix has $(2 \times L)$ and a multivariate Gaussian with full covariance matrix $\left(2 \times L + \frac{L \times (L-1)}{2}\right)$ free parameters. In other words, lHMM will require a substantially fewer number of parameters than a multivariate Gaussian with full covariance matrix and, whenever $2M < 3L$, a lHMM has fewer parameters than a Gaussian with diagonal covariance matrix. Furthermore, the lHMM captures low order dependencies, whereas the Gaussian with diagonal covariance



(a) (b) (c)

Figure 4.7: *We depict heat map plots of the absolute value of the empirical covariance of data sampled from a 19 state lHMM from the up-regulation lHMM depicted in Figure 4.1, where the up-regulation event occurred before time point 7 (a), between 7 and 12 (b), and time courses after time point 12 (c). Dark blue entries indicates zero values, while red indicates positive values.*

matrix assumes independence between time points. Of course, one could estimate the full covariance matrix, but as discussed in Section 2.3.4, such models could present over-fitting problems when used in mixture models. Thus, due to its characteristics, lHMMs are an ideal candidate for component models in a mixture model for analyzing gene expression time courses, as it requires few free parameters and is able to model temporal dependencies.

4.3 Querying of Time Courses

An initial exploratory analysis of gene expression time course can be performed by querying a particular data set for specific patterns of expression. This can be done by specifying a HMM topology and parameters manually and, later, ranking all time courses in $\mathbf{X}$ by their likelihood for that particular model (Eq. 4.7).

More formally, for a given linear HMM θ^l, a data set $\mathbf{X}$ and a stringency value $v < N$, a query can be described as follows

$$\mathbf{S} = \{x | \text{rank}\left(p(x|\theta^l)\right) \leq v, x \in \mathbf{X}\} \tag{4.17}$$

where $p(x|\theta^l)$ is the likelihood function defined as in Eq. 4.7, the function rank returns the rank of ordering x in relation to the likelihood, and $\mathbf{S}$ is the set of expression profiles, which have highest v likelihoods under the model θ^l.

As the knowledge discovery process in the analysis of biological data is human-centric, a high degree of interactivity is important in an initial analysis. We develop an interactive tool—the Graphical Query Language (GQL)—for querying data sets from gene expression time courses [53]. The tool allows the user to define a linear HMM model, tune its parameters and to define the query stringency (v) interactively. Modifications of the linear HMM parameters in the interface tool are simultaneously showed in the time courses

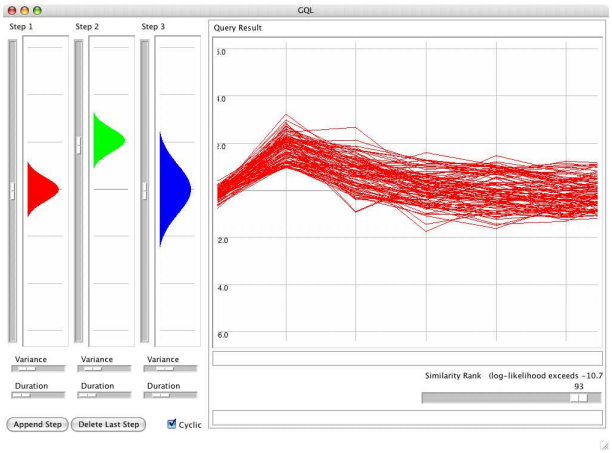


Figure 4.8: *The GQLQuery interface is divided into two main components: the left part is the model editor, where the user can view and change the model's parameters, and in the right part is the query result, where the queried time courses are displayed.*

panel, which plots the time courses on $\mathbf{S}$. The user can explore all possible prototypical patterns of expression presented in the data set. By scrolling the mouse over a desired time course, GQLQuery will depict the Viterbi path in the bar below the time course plots, as well as display information as the gene name. The interface also allows the user to create models for periodic time courses. All query results and generated models can be saved and used for further analysis.

4.4 Mixture of linear HMMs

In order to find groups of co-expressed time courses, we combine K linear HMMs in a mixture model. With such a model, we can find groups of expression patterns displaying a similar prototypical behavior within a gene expression data set. Formally,

$$p(x|\Theta) = \sum_{k=1}^{K} \alpha_k p_k(x|\theta_k^l), \tag{4.18}$$

where α_k is the mixture coefficient respecting $\alpha_k \geq 0$ and $\sum_{k=1}^{K} \alpha_k = 1$ (see Section 2.2), $p_k(x|\theta_k^l)$ is the density corresponding to the kth HMM as defined in Eq. 4.7, and $\Theta = (\alpha_1, ..., \alpha_K, \theta_1^l, ..., \theta_K^l)$.

We can estimate mixtures with the use the EM algorithm described in Section 2.3.1, by the inclusion of a missing variable Y, where $y \in \{1, ..., K\}$ indicates which mixture component produced the observation x. As previously discussed, the only necessary extension is the specification of the M-Step. In the M-Step, we need to obtain for a particular cluster assignment the maximum likelihood estimate of each of the models θ_k^l. While there is no analytic way for computing the maximum likelihood estimate for HMMs, we can apply the

Baum-Welch algorithm to (locally) maximize Eq. 4.8 in an iterative procedure.

Note that there is no need to extend the Baum-Welch algorithm for the specific topology of linear HMMs. All zeros entries in the transition matrix A, which are imposed by the linear topology, will remain equal to zero after the application of the Baum-Welch [121], and the topology will always be preserved. The parameter ϕ, related to the missing values, is kept fixed.

4.4.1 Model Initialization

The EM algorithm requires an initial model as starting point for the inference. This can be performed with random assignments, as described in Section 2.3.1. The lHMMs require the previous specification of few parameters, for example, the number of states M for each θ_k^l. Note that each component model can have a distinct number of states as long as $M \leq L$. Also, in the initialization of the duration parameters the property $\sum_{u=1}^{M} d_u = L$ should be respected, where L is the length of the time courses (see Section 4.2.1 for restrictions for the periodic extension of lHMMs). We will refer to this initialization method as the random model collection (`RMC`).

Another alternative is to use the results of other clustering methods as the initial assignment for the mixture [145]. Thus, we also consider the use of k-means for initializing the models, which we will reefer to as (`KMC`).

The inference of HMM topology can be solved for special topological cases with the use of priors and additional computational cost [61, 184, 205]. In a previous work [185], a collaborator proposed an adaptation of the Bayesian model-merging algorithm [205] to continuous emissions and linear HMMs. In this context, [185] takes advantage of the linear topology of the lHMM. It starts with a topologically unconstrained maximum likelihood model, which has one linear HMM per gene-expression time course x (or a branch), and one state per time-point in each branch, i.e., $L = M$. The method in [185] is based on merging states of the unconstrained model in two steps.

First, it merges states within the branches. The method identifies for each branch states whose merging decreases the likelihood the least. As the merging method assumes that variances are equal for all states, it only has to identify those successive states whose means are similar. A merging of components is performed until it reaches the desired number of states. Typical values range from a third to half the number of time-points. Second, the method merges the shrunken branches such that the loss of likelihood is minimal in each merging step with hierarchical clustering. For detailed description of the algorithm, which we will refer to as Bayesian Model Collection (`BMC`), see [185].

4.4.2 Viterbi Decomposition

The idea of the Viterbi decomposition is to find sub-groups of synchronous groups within genes following the same asynchronous prototypical patterns as modeled by a lHMM. An example on how the Viterbi path can be used in finding asynchronous groups is depicted in Figure 4.4. In that simple example, all time courses presented only three distinct paths. On real data and with a lHMM with more states, the number of distinct Viterbi paths is larger, and simple path enumeration would lead to a high number of asynchronous groups. One way to overcome this is to apply a clustering method for finding groups of similar Viterbi paths.

For a set of time courses x belonging to a cluster k, or $\mathbf{X_k} = \{x|y = k, x \in \mathbf{X}\}$, we measure the most probable Viterbi path q of each of the time courses $x \in \mathbf{X_k}$

$$\mathbf{Q_k} = \{q|q = \arg\max_q p(x, q|\theta), x \in \mathbf{X}_k\} \tag{4.19}$$

and look for groups of similar Viterbi sequences q_i.

In the approach proposed by a collaborator [132, 185] and used in the following analysis, the main assumption is that the sequence paths over one state will be the most significant in finding a group of synchronous time-courses. Take, for example, a group of cyclic time courses. Sorting these according to the time at which the first peak is reached is enough to produce subgroups of time courses with the same phase shift. Initially, the method searches the most appropriate state by calculating a Silhouette coefficient [178] for all states, and returns the one with lower Silhouette, i.e., the one with more compact and isolated sub-groups. After one state is selected, all possible unique paths are enumerated, and a greedy clustering method is performed by joining pairs of sub-groups leading to the highest increase in the Silhouette coefficient, until no more increase is possible. See [132, 185] for details of the method.

4.4.3 GQLCluster

All methods described in this section are implemented in the tool GQLCluster [53]. This software also includes some additional features useful in the analysis of time courses, such as methods for filtering gene expression data sets, an implementation of standard clustering methods such as k-means and hierarchical clustering, mixture of Gaussians and a semi-supervised approach described in [187].

After clustering by mixtures of lHMMs (or other clustering methods) has been carried out, GQLCluster offers several interactive tools for the analysis of the results. As a starting point, the graphical interface creates panels, which contain the time courses of each of the clusters/components (Figure 4.9). Then, for each cluster, it is possible to inspect the list of gene identifiers, which are linked to known web databases. In the mixture model methods, the time courses are assigned to the most likely model. The user can choose

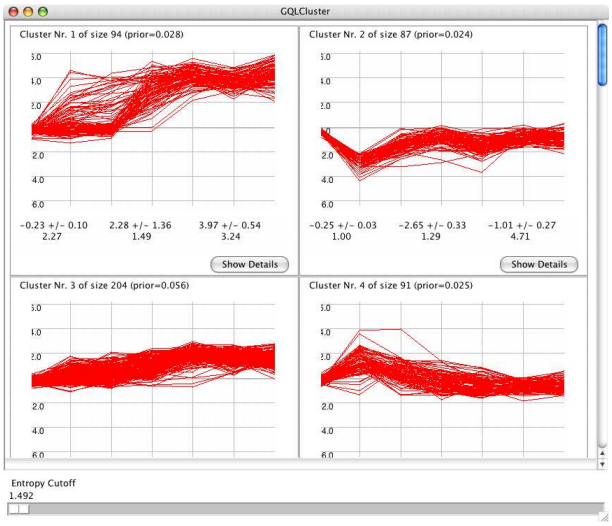


Figure 4.9: *After estimation, GQL Cluster displays the time courses assigned to each cluster. The user can then perform a more detailed inspection of the modules, such as looking for gene annotation in known web databases, inspection for GO enrichment or computation of sub-groupings.*

only genes that can be unambiguously assigned to one model by increasing the entropy cut-off threshold. A further refinement of the clusters can be obtained by the application of a Viterbi decomposition analysis, which finds sub-groups of synchronous time courses. All results, plots and estimated models can be saved for further analysis. A tutorial on the application, data and installation instructions can be found at [90].

4.5 Experiments

In order to illustrate the use of all the methods proposed in Section 4.2.1, we describe in the following the computational experiments based on two gene expression time course data sets. For benchmarking purposes, we make use of fully annotated data from yeast cell cycle. With this data set, we make a comparison with other clustering methods such as clustering with Splines [14], CAGED [172] and k-means. We also test the effects of initialization procedures: random model initialization (`RMC`), k-means initialization (`KMC`) and Bayesian initialization (`BMC`) (see Section 2.3.2 for details). Furthermore, we also test the use of the Viterbi decomposition (`VD`) for finding sub-groups in the clusters. For the second data set, which is based on a human HeLa cell cycle, we evaluate the use of the Viterbi decomposition and the entropy threshold proposed in Section 3.1. For the evaluation of the entropy threshold method, we make use of a specificity score based on the functional annotation of genes.

4.5.1 Data

Yeast Cell Cycle `YCC`**.** This data represents the expression level of around 6000 genes during two cell cycles from yeast measured in 17 time points [42]. As in [233], we used a subset of this data in the experiments. This data set, 5-phase criterion, abbreviated as `YCC`, contains 384 genes visually identified to peak at five distinct time points [42], each representing a distinct phase of cell cycle. The expression values of each gene are standardized. As showed in [233], such a procedure enhances the performance of model-based clustering methods, when the original data consists of intensity levels. This data set has class labels for the full range of genes, allowing the comparison of distinct methods.

HeLa Cell Cycle. We use published data from a time course experiment, in which the authors measured genome wide gene expression of synchronized HeLa cells [226]. We use 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, are measured every hour from 0 to 46 hours. For reasons of comparison, we exclude clones showing missing values from further analysis. We obtain log ratios by dividing all expression values by the reference value (time point 0h) and taking the logarithm. Furthermore, the data is pre-processed by extracting all those genes with an absolute fold change of at least two in at least one time point. This results in a data set containing 2,272 expression time courses.

Specificity Score based on Gene Annotation. Gene Ontology (GO) describes genes in three distinct categories [9]: cellular component, molecular function and biological process. GO has a 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 process 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 relation between ambiguity of gene clusters and the specificity (or level) of GO annotations (see Section 6.2.2 for a more detailed description of GO).

In order to find GO annotations related to a given subset of genes, we use an enrichment analysis [24], to look for annotation terms that are over-represented in this subset. The probability that this over-representation is not found by chance can be measured with the use of a hyper-geometric Fisher exact test [199]. The enrichment test returns for each cluster and GO term a p-value describing how statistically significant is a particular GO term for describing genes in a particular cluster[3]. See Appendix A for a description of the test.

[3] In the subsequent chapters, when a particular GO term is over-represented for a given cluster, we state GO Term X is enriched in cluster Y, or we found enrichment for GO Term X in cluster Y

Table 4.1: *Results of the different methods on* YCC.

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

The calculation of the specificity (or level) of the annotations from a set of genes is straightforward. Given a cluster, we repeat the enrichment test for each term in GO, and retrieve the ones exceeding a given p-value. Then, the length of the path from the root to each enriched GO term is counted and averaged. Since GO is a DAG, one node can be reached by more than one path from the root. Therefore, the average of all possible path lengths is taken. The final score reflects the mean distance of the enriched GO terms to the root of the Gene Ontology. The larger the score value, the higher is the specificity of the functional annotations enriched in the evaluated gene clustering.

4.5.2 Results

Yeast Cell Cycle. As Table 4.1 illustrates, the k-means algorithm obtains a good result with a corrected Rand (CR) of 0.43 (specificity of 0.54 and sensitivity of 0.56). Mixture estimation with BMC and a posterior Viterbi decomposition obtains the highest values for all indices, with a CR of 0.467, specificity of 0.55 and sensitivity of 0.66. The results of CAGED are not included, since it could only find one cluster, which makes the calculation of the indices impossible. Note that the number of clusters in CAGED cannot be controlled by the user. Moreover, model-based clustering with splines, another method taking temporal dependencies into account, has an overall poor result. Furthermore, the mixtures of HMMs (HMM Mix.), which perform "soft" cluster assignments during the EM, have a better performance than the use of model-based clustering with HMMs (HMM Clu.), which performs hard assignments during the EM method.

Looking at the results in more details, we find that most methods join genes from the cell cycle class"Late G1" with "S" and genes from the cell cycle class "M" with "Early G1".

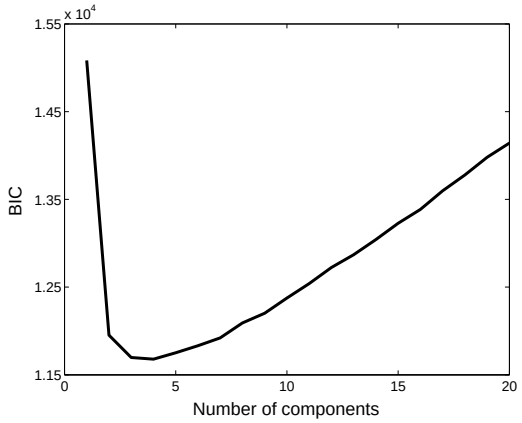


Figure 4.10: *BIC versus number of components for the data set* YCC*. The correct number is five.*

Note that these classes correspond to cell cycle phases with a small phase-shift and their genes have very similar time courses. Interestingly, the BIC index underestimate the number of components by one in YCC (Figs. 4.10), which is a further indication of the difficulty of separating these two classes. The application of the Viterbi decomposition after estimation of mixtures of HMMs improves the results, in particular for these "difficult" classes, which indicates the usefulness of the Viterbi decomposition in refining the clusters.

In relation to the initialization method, BMC obtains higher accuracy than KMI and RMC in most of method combinations. 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, in contrast to use of RMC or KMI.

HeLa Cell Cycle. The initial collection used for mixture estimation for this data set consists of 35 lHMMs with 24-states obtained from RMC. Two groups have periodically expressed time courses. We apply the Viterbi decomposition to these groups. For cluster 1 (Figure 4.11), the first subgroup contains 26 genes known to be in cell cycle phase G2 and one gene to cell cycle phase G2/M, the second subgroup eleven G2 and 19 G2/M, the third subgroup 31 G2/M, two M/G1 and 1 G1/S (see Section 4.1 for description of cell cycle phases). The second group (not shown) contains twelve G1/S and two S-phase genes. Both CDC2 representatives are found in the same subgroup (Figure 4.11, phase 1). Furthermore, cyclin A (Figure 4.11, phase 2) and cyclin B (Figure 4.11, 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 [226]. The same holds for the G1/S, S phase subgroups.

In relation to the entropy threshold, we observe an increase in the GO specificity score for lower entropy threshold values, followed by a decrease of specificity for very low threshold values (see Figure 4.12). The mean GO specificity score raises considerably (around 2.0)

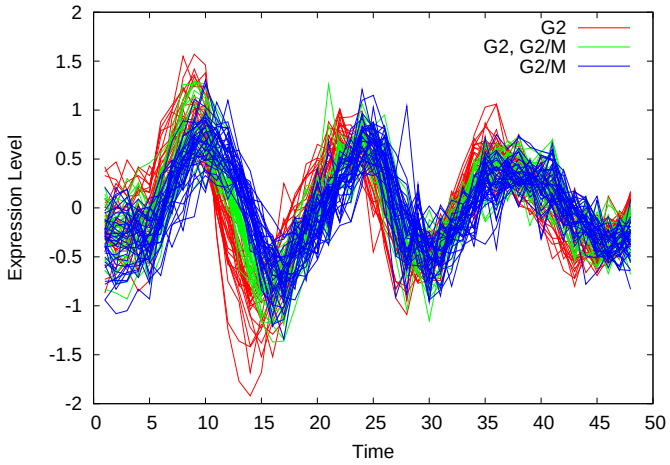


Figure 4.11: *One group has periodically expressed time courses in the* `HeLa` *data set. This group is subsequently decomposed into three subgroups (top to bottom), corresponding to groups of synchronous genes, with the Viterbi decomposition. The first subgroup (red) contains mainly cell cycle phase G2 genes, the second (green) G2 as well as G2/M genes and the third (blue) mostly G2/M genes.*

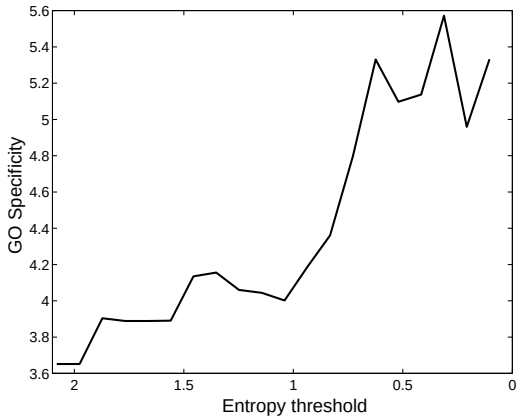


Figure 4.12: *The mean GO specificity level of clusters from* `HeLa` *versus the entropy threshold. The lower the threshold, the less unambiguous are the cluster assignments.*

until the threshold value 0.3. This result indicates that less ambiguous assignments, as derived by the posterior probabilities returned by the mixture model, lead to an enrichment of more specific GO annotations. Therefore, the entropy threshold is a valuable tool for refining the results returned by the mixture of lHMMs and deriving fine grained groups of functionally related genes.