

# A Sequential Monte Carlo Sampling Approach for Cell Population Deconvolution from Microarray Data

Sushmita Roy<sup>§</sup>, Terran Lane<sup>§</sup>, Chris Allen<sup>†</sup>, Anthony D. Aragon<sup>†</sup>, Margaret Werner-Washburne<sup>†</sup>

<sup>§</sup> Department of Computer Science, University of New Mexico, Albuquerque, NM 87131

<sup>†</sup> Department of Biology, University of New Mexico, Albuquerque, NM 87131

## Abstract

Microarray analyses assume that gene expressions are measured from synchronous, homogeneous cell populations. In reality, the measured gene expression is produced by a mixture of populations in different stages of the cellular life cycle. Hence, it is important to estimate the proportions of cells in different stages and to incorporate this information in the analysis of gene expression, clusters, or function. In this paper we propose a novel unsupervised learning approach that models biological processes, such as the cell-cycle, as a hidden state-space model and uses a particle-filter based approach for estimating model parameters. Our approach finds a maximum a posteriori (MAP) estimate of the cell proportions given the gene expression and an estimate of the stage dependant gene expression. Evaluation of statistical validity of our approach using randomized data tests reveals that our model captures true temporal dynamics of the data. We have applied our approach to model the yeast cell-cycle and extracted profiles of the population dynamics for different stages of the cell-cycle. Our results are in agreement with biological knowledge and reproducible in multiple runs of our algorithm, suggesting that our approach is capable of extracting biologically meaningful and statistically significant information. Finally, the stage dependant gene expression can be used to determine clusters of active genes on a per-stage basis.

## 1 Introduction

Gene-expression analysis of linear or cyclic biological processes is generally carried out with what are assumed to be synchronous populations of cells. That is, it is assumed that the observed gene expressions are drawn from a homogeneous population of cells, all in the same stage of the cellular life cycle at any given time point. A variety of synchronizing techniques have been used to arrest populations in a particular stage of the cell cycle [7], including conditional mutants and nutrient starvation. In practice, however, natural populations of cells at any point in time are heterogeneous. This is because, firstly, it is nearly impossible to achieve *perfect* synchronization – there are almost always some cells in the population that fail to arrest in the synchronizing stage; and secondly, even if all cells do arrest in the same stage, they typically re-enter and proceed through the cell cycle at different rates, resulting in the rapid loss of synchronization [3]. As a result, for most microarray experiments, measured gene expression is actually a mixture of expressions from multiple populations of cells. The ability to determine the relative abundance of each cell population in a sample would simplify the analyses of mixed populations of cells not only by providing a reliability measure for each dataset and a description of the population dynamics over time but also for determining population distributions for cultures under a variety of other conditions [13]. The problem of studying the cell population dynamics by estimating the proportion of cells in different stages during the course of a microarray experiment is called Cell Population Deconvolution (CPD).

In this paper, we propose a new, data-driven, probabilistic approach to the CPD problem by formulating the biological process as a temporal hidden state identification problem. The challenge here is that, unlike in well-understood systems such as discrete hidden Markov models (DHMMs) [19, 20], the CPD hidden state at each time point is a *multinomial vector* representing the fraction of cells in each stage at that time. This crucial difference allows us to recover the dynamics of evolving populations of cells over time, but yields difficult inference and parameter estimation (learning) problems. To render these problems tractable we (a) assume a Dirichlet posterior distribution over the hidden state at each time point [16]; (b) assume that individual gene expressions are independent and Gaussianly distributed, given the hidden state; (c) assume first-order Markov transition dynamics for the hidden state; and (d) employ a particle filtering-based algorithm for hidden state distribution inference [1, 12, 5] and the Expectation Maximization (EM) algorithm for parameter estimation [4]. With this approach we are able to estimate cell population distributions over time (hidden state), uncertainty over the hidden state (hidden state posterior distribution), the dynamics of the biological process (process or transition model), and the expected gene expression values for each stage of the cell cycle (observation or output model) purely from observed gene expression data.

Using yeast cell cycle microarray data [21], we demonstrate that our approach recovers high-quality estimates of the temporal evolution of cell population distributions. Although our algorithms are purely data-driven and do not rely on prior knowledge of cell cycle dynamics or gene expressions for particular cell stages, the recovered temporal information for the different populations are in agreement with the known ordering of cell-cycle stages and the deconvolved gene expressions correspond well with the expressions of known stage-specific genes.

An alternate computational approach to this problem [13] attempts to deconvolve a mixed population into the individual stages of the cell cycle. The observed gene expression vector is considered to be a linear combination of pure gene expression vectors for stages of the cell cycle, weighted by the proportion of cells in each stage. The pure gene expression vector for a cell-cycle stage is identified from existing microarray data [21], on the basis of the expression level of a specific gene, called a marker gene, known to be required for that particular stage. The timepoint at which the marker gene has the highest expression, is considered as the “pure” expression vector for the corresponding cell-cycle stage assuming that perfect population synchronicity is maintained at these time points. Reliance on either perfect synchronization or on prior knowledge of gene expressions, however, is quite limiting in practice. A purely data-driven approach that does not assume perfect synchronization at any time point is more flexible and biologically realistic.

Non-computational methods have also been used to provide measures of population synchronicity. *Fluorescent activated cell sorting* (FACS) allows qualitative analysis of population synchrony by measuring the genetic content of the cells in the population [18]. Another method uses image analysis of budding yeast images [17]. However, these approaches are expensive and time consuming and need to be carried out at the time when cells are harvested. A probabilistic approach is more robust, scalable and general.

Overall, our model has several advantages: it provides a probabilistic framework which handles noisy data efficiently, incorporates temporal dependencies that are inherent to time-series data, directly estimates the population fractions at every time-point, estimates the stage specific gene expression, provides a clustering technique for genes with similar expression and finally does not make any assumptions of linearity in the process dynamics.

## 2 A hidden state space model for CPD

### 2.1 Model overview

Let  $P$  denote a large population of cells used in a time-series microarray experiment having  $T$  time points. Let  $b$  be the temporal biological process under study using  $P$ .  $b$  is assumed to be an ordered set of  $n$  biological events or stages, denoted as  $bs_k$  for  $1 \leq k \leq n$ . E.g. for the cell-cycle, we take the  $n = 5$  stages

from the set  $L = \{M, M/G1, G1, S, G2/M\}$  (described in detail in Section 4.1). We note that even though  $n$  is predetermined, the mapping between a  $bs_k$  and an element of  $L$  is not known. At any time point  $t$ , where  $1 \leq t \leq T$ , a cell in  $P$  can be in one of these  $n$  stages. Thus at every  $t$ ,  $P$  is composed of  $n$  different fractions of cell populations, one in each  $bs_k$ . Let  $x_t$  denote the hidden state vector, which defines  $P$ 's fractions at  $t$ , with the component  $x_t(k)$  representing  $P$ 's population fraction in the  $bs_k$  stage. Therefore,  $x_t$  is a multinomial vector obeying the constraints  $0 \leq x_t(k) \leq 1; \forall k$  and  $\sum_k x_t(k) = 1$ . The behaviour of  $b$  is observed by gene expression vectors,  $y_t$ . The  $m$  components of  $y_t$  are the relative expression levels of the  $m$  individual genes, denoted  $y_t(j)$  for  $j = 1, \dots, m$ . The Cell Population Deconvolution problem, then, is: given a time series of microarray observation vectors,  $\{y_1, \dots, y_T\}$ , estimate the hidden fraction of cell stage occupancies at every time point,  $x_1, \dots, x_T$ .

We assume that the observed gene expressions,  $y_t$ , are conditionally dependent only on the corresponding hidden cell stage distribution specified by  $x_t$ , and that  $x_t$  itself is conditionally dependent only on the preceding cell stage distribution  $x_{t-1}$ . This yields a temporal process similar in statistical structure to a hidden Markov model, with a first-order Markov *process model*,  $P(x_t|x_{t-1})$  and an *observation model*  $P(y_t|x_t)$ . Unfortunately, the fact that the hidden variable is a multinomial vector rather than a discrete scalar complicates hidden state inference and parameter learning considerably. The natural choice for prior or posterior distribution over multinomial vectors,  $P(x_t)$  or  $P(x_t|y_1, \dots, y_t)$ , is a Dirichlet distribution, but it is not clear what form a process model for a Dirichlet should take, nor how to carry out inference on such a model.

In this paper, we apply a sequential Monte Carlo sampling method based on particle filters to the state inference problem. We assume the observation model to be described by a set of Gaussian parameters,  $\mu_k^j$  and  $\sigma_k^j$ , for the gene expression of the  $j^{\text{th}}$  gene in the  $k^{\text{th}}$  stage. We assume the process model to be specified by a transition matrix  $A$ . Finally, we assume that the posterior hidden state distribution for every  $t$ ,  $P(x_t|y_1, \dots, y_t)$  is specified by a Dirichlet  $D_t$ . Thus our complete model for  $b$  is specified by three sets of parameters: stage dependant Gaussian parameters for the gene expression; the transition matrix; and the posterior hidden state distribution. These parameters are learnt by our parameter estimation procedure based on Expectation Maximization (EM). Our training algorithm uses a ‘‘forward step’’ to estimate  $P(x_t|y_1, \dots, y_t)$  and a ‘‘backward step’’ to estimate  $P(x_t|y_{t+1}, \dots, y_T)$ . Combining these two estimates we obtain the transition probabilities in  $A$  (refer to Sections 2.3 and 3.3). We use maximum likelihood estimates to obtain the Gaussian parameters for the stage dependant gene expressions. The steps are described in more detail in Section 3.

After running the EM procedure to convergence, we have a complete model of the biological process  $b$ , given the observed microarray time series. The model parameters inform us about the process dynamics ( $A$ ), the stage-specific gene expression profiles ( $\mu_k^j$  and  $\sigma_k^j$ ), and the hidden state of cell population distributions ( $D_t$ ). The latter gives the solution of our original deconvolution problem: we take the desired population distribution estimate for time  $t$  to be the expected value of the hidden state posterior,  $P(x_t|y_1, \dots, y_t)$ , specified by  $D_t$ . Different components of our model are detailed in the following sections.

## 2.2 Particle filters for state space models

Seeking a closed form for the inference equations to model systems like  $b$ , maybe intractable. Particle filters can be used to get around this problem by estimating the hidden state probabilities using a set of weighted samples. We use ‘‘samples’’ and ‘‘particles’’ interchangeably. Samples are drawn from an *importance density*  $q$ , and sample weights are used to determine the likelihood of the observed data given the sample. Our model uses a *Sample Importance Resampling* (SIR) filter [8] which simulates a temporal process in the following way:

1. Generate a set of samples for the first time point,  $x_1^i$  from the importance density,  $x_1^i \sim q(x_1)$

2. Calculate sample weights,  $w_t^i$ , using the observation model,  $w_t^i = P(y_t|x_t^i)$ .  $w_t^i$  is proportional to the likelihood of the observed data  $y_t$
3. Normalize the weights and resample the particles. The resampled particles estimate the posterior  $P(x_t|y_1, \dots, y_t)$ .
4. Project the samples to the next time step, using the process model, yielding  $x_{t+1}^i \sim P(x_{t+1}|x_t)$
5. Repeat 2–4 for  $t = 2 \dots T$

Our particle filtering algorithm uses a slight variant of the SIR filter as described in Section 2.7

### 2.3 Process model

The process model, gives the prior estimate of the next state, i.e.  $P(x_{t+1}|x_t)$ . This is determined by the probability that a cell fraction in  $bs_k$  at  $t$  transitions to  $bs_l$  at  $t + 1$  i.e.,  $P(bs_l(t + 1)|bs_k(t))$ , where  $1 \leq (k, l) \leq n$ . These transition probabilities are specified by the transition matrix,  $A$ . Every element,  $A(lk)$ , specifies the probability  $P(bs_l(t + 1)|bs_k(t))$ , where  $l$  and  $k$  represent the rows and columns of  $A$  respectively. Therefore,  $\sum_{l=1}^n A(lk) = 1$ ,  $1 \leq k \leq n$ .

### 2.4 Observation model

The observation model gives the estimate of the current observation,  $y_t$ , given the current state,  $x_t$ , i.e.  $P(y_t|x_t)$ . We make the following assumptions for the observation model:

- The measured expression of the  $j^{th}$  gene,  $g_j$ , in  $bs_k$ , where  $1 \leq j \leq m$ , follows a Gaussian distribution,  $N(\mu_k^j, \sigma_k^j)$ . Since the actual population from which mRNA is extracted is a mixture of cell-populations in different  $bs_k$ , the observed gene expression is from a mixture of Gaussians. Hence,  $P(y_t(j)|x_t) = \sum_{k=1}^n x_t(k)P(y_t(j)|\mu_k^j, \sigma_k^j)$ , where  $y_t(j)$  is the expression value of  $g_j$  at  $t$ .  $x_t(k)$  is the contribution of the  $k^{th}$  Gaussian from the population fraction in  $bs_k$ .
- The expression value of one gene is independent of another at the same timepoint given a biological stage. Hence, the probability of seeing an entire gene expression vector,  $y_t$  can be written as a product of probabilities of individual gene expressions, i.e.,  $P(y_t|x_t) = \prod_{j=1}^m P(y_t(j)|x_t)$ . Therefore, the  $P(y_t|x_t)$ , can be written as  $P(y_t|x_t) = \prod_{j=1}^m \sum_{k=1}^n x_t(k)P(y_t(j)|\mu_k^j, \sigma_k^j)$

### 2.5 Importance density $q$

Since  $x_t$  is a multinomial distribution, the obvious choice of the importance density  $q$  for our model is a Dirichlet. Moreover, choosing a Dirichlet has the added advantage that its parameters can be assigned values to reflect prior knowledge of the cell fractions in the different biological stages.

### 2.6 Sample weight calculation

Sample weight  $w_t^i$ , of  $x_t^i$ , is given by the observation model, i.e.  $P(y_t|x_t^i) = \prod_{j=1}^m \sum_{k=1}^n x_t^i(k)P(y_t(j)|\mu_k^j, \sigma_k^j)$ . This is similar to  $P(y_t|x_t)$  in Section 2.4, with  $x_t$  being replaced by  $x_t^i$ . The rationale for replacing  $x_t$  by  $x_t^i$  is that  $x_t^i$  represents a possible description of the underlying population, which is exactly described by  $x_t$ .

### 2.7 Modifications to the SIR filter

One of the problems that we encountered with the SIR filter was rapid sample degeneration. This was because a small subset of the entire sample set had very high weights compared to the rest. The resampling step repeatedly selected these samples, resulting in the degeneration of all or most of the samples to single

points. The true probability distribution cannot be represented by such few samples and Dirichlets cannot be estimated from these samples due to numerical instability. Hence we propose two modifications that incorporate smoothing strategies to overcome the sample degeneration problem.

The first modification is in the *sample projection* step. Unlike the traditional approach, wherein samples are generated only at  $t = 1$ , and projected forward for all  $t > 1$ , we generate a new set of samples at every  $t$ . Let  $\{\hat{x}_t^1, \hat{x}_t^2, \dots, \hat{x}_t^N\}$  denote sample set representing the prior,  $P(x_t|x_{t-1})$ . These samples are obtained by projecting samples from  $t - 1$  via  $A$ . We calculate the normalized weights,  $\{\bar{w}_t^1, \bar{w}_t^2, \dots, \bar{w}_t^N\}$ , where  $\bar{w}_t^i = \frac{w_t^i}{\sum_{s=1}^N w_t^s}$ . Then we resample using the normalized weights and re-estimate a new Dirichlet  $D_t$  from the samples chosen by the resampling step.  $D_t$  is then sampled to generate a new set,  $\{x_t^1, x_t^2, \dots, x_t^N\}$ , that represents  $P(x_t|y_t)$ . These samples are then projected to  $t + 1$  via  $A$  to result in  $\{\hat{x}_{t+1}^1, \hat{x}_{t+1}^2, \dots, \hat{x}_{t+1}^N\}$ .

The second modification is in the *resampling* step. Let  $S_t$  denote the sample set resulting after resampling at time  $t$ . To avoid samples from collapsing to a single point, we perturb every  $\hat{x}_t^i$  selected by the resampling step to result in  $\tilde{x}_t^i$  and add  $\tilde{x}_t^i$  to  $S_t$ . The perturbation is done by adding a sample from a Gaussian,  $N(0, \omega)$ , to every element of  $\hat{x}_t^i$ , i.e.  $\tilde{x}_t^i(k) = \hat{x}_t^i(k) + \nu$ , where  $\nu \sim N(0, \omega)$ ,  $\forall k, 1 \leq k \leq n$ .  $\omega$  controls the extent of perturbation. Thus if  $\hat{x}_t^i$  were selected  $p$  times, it would be represented by the set  $S_t^i = \{\hat{x}_t^i, \dots, \hat{x}_t^i\}$  and  $S_t^i \subset S_t$ . By adding a Gaussian perturbation, the constraint that  $\hat{x}_t^i(j)$  should be a multinomial, maybe violated. To enforce this constraint we keep iteratively perturbing  $\hat{x}_t^i$  and choose only those  $\tilde{x}_t^i$  such that  $\tilde{x}_t^i(k) \geq 0$ ,  $\forall k, 1 \leq k \leq n$ . The selected  $\tilde{x}_t^i$  is then normalized such that  $\sum_{k=1}^n \tilde{x}_t^i(k) = 1$  to result in a multinomial. The value of  $\omega$  is chosen by experimentation.

### 3 Model training and parameter estimation

Based upon the model description in Section 2.1, the parameters of the model for  $b$  are described by the tuple  $\Theta = \{A, \mu_k^j, \sigma_k^j, D_t\}$ . Parameter estimation is done using a learning procedure based on Expectation Maximization (EM). Our learning procedure is similar to the Baum-Welch algorithm for HMM [19]. The hidden variables in the system are the state variables  $x_t$ . During the Expectation (E) step we use the model parameters and obtain the expected values for  $x_t$ . During the Maximization (M) step we use the expected values of  $x_t$  and estimate the model parameters.

#### 3.1 Forward and backward variables

Before stepping into the details of the E and the M steps, we define a set of variables which are similar in concept to those used in the Baum-Welch algorithm. The difference in our algorithm is that all the variables are multi-dimensional.

- $\alpha_k(t)$  is the probability of being in  $bs_k$ , at time  $t$ , given observations  $\{y_1, \dots, y_t\}$ .  $\alpha_k(t)$  is specified by  $x_t^i(k)$ .
- $\beta_k(t)$  is the probability of being in  $bs_k$ , at time  $t$ , given observations  $\{y_{t+1}, \dots, y_T\}$ .  $\beta_k(t)$  is specified by  $xb_t^i(k)$ .  $xb_t^i$  represents a sample from the backward step at time  $t$ .
- $e_k(t)$  represent the probability of seeing the observation  $y_t$  from  $bs_k$ . This is just the product of the probabilities of individual genes expressions from  $bs_k$ , due to the second assumption in the observation model (refer Section 2.4).
- $\gamma_k(t)$  is the probability of being in  $bs_k$ , at time  $t$ , given all observations,  $\{y_1, \dots, y_T\}$ . Similar to HMM,  $\gamma_k(t)$  is given by  $\gamma_k(t) = \frac{\alpha_k(t)\beta_k(t)}{\sum_{l=1}^n \alpha_l(t)\beta_l(t)}$ .
- $\xi_{lk}(t)$  is the probability of being in  $bs_l$ , at time  $t$  and in  $bs_k$ , at time  $t + 1$ , given  $\{y_1, \dots, y_T\}$ . Similar to HMM,  $\xi_{lk}(t)$  is given by  $\xi_{lk}(t) = \frac{\alpha_l(t)A(kl)\beta_k(t+1)e_k(t+1)}{\sum_{q=1}^n \sum_{r=1}^n \alpha_q(t)A(rq)\beta_r(t+1)e_r(t+1)}$ .

### 3.2 E step

The *E* step is made up two parts: *forward* step and *backward* step. The samples for the first time-point,  $t = 1$ , are drawn from a Dirichlet with parameters to reflect our belief of the cell population. E.g., if we believe that the population was synchronized and majority of the cells are in the  $k^{th}$  biological stage, then the  $k^{th}$  parameter can be set to 0.8 and the remaining parameters to 0.2. Based upon the traditional SIR filter and our modifications proposed in section 2.7, the forward step proceeds as follows:

1. Generate samples,  $\{\hat{x}_1^1, \dots, \hat{x}_1^N\}$ , from  $\hat{D}_1$ , whose parameters are set on prior knowledge about the synchrony of the cell population. These samples represent the prior estimate  $P(x_1)$ .
2. Calculate sample weights using (2). Set  $t = 1$ .
3. Normalize weights and resample using the second modification in section 2.7 to result in  $\{\bar{x}_t^1, \dots, \bar{x}_t^N\}$ .
4. Estimate  $D_t$ , using  $\{\bar{x}_t^1, \dots, \bar{x}_t^N\}$ .
5. Generate samples,  $\{x_t^1, \dots, x_t^N\}$ , from  $D_t$ . These samples represent the posterior  $P(x_t|y_1, \dots, y_t)$ .
6. Project  $\{x_t^1, \dots, x_t^N\}$ , to the next time step to result in  $\{\hat{x}_{t+1}^1, \dots, \hat{x}_{t+1}^N\}$ , which estimate the prior  $P(x_{t+1}|x_t)$ , where  $2 \leq t \leq T$ , where  $T$  is the total number of time-steps in the data.
7. Repeat steps from 2 to 6 with  $t$  ranging from  $\{2, \dots, T\}$ .

The backward step is used to calculate  $\beta_i(t)$ . We begin with the samples from a uniform Dirichlet, for  $\hat{D}_T$  i.e. all parameters are equal to 1. The backward step proceeds as follows:

1. For every  $xb_{t+1}^i$ ,  $xb_t^i(j) = \sum_{k=1}^n xb_{t+1}^i(k)A(kj)e_j(t+1)$ ,  $\forall 1 \leq j \leq n$ .
2. After all the components  $xb_t^i$  are calculated, it is normalized to result in a multinomial. These samples represent  $P(x_t|y_{t+1}, \dots, y_T)$ .  $xb_t^i(k)$  represents  $\beta_k(t)$ .
3. Calculate weights for  $xb_t^i$  followed by resampling and reestimation of the backward Dirichlet and generation of a new set of samples from the newly estimated Dirichlet as described in the forward step.
4. This process is repeated for all the timesteps.

### 3.3 M step: Estimation of A

Every entry  $A(lk)$  specifies  $P(bs_l(t+1)|bs_k(t))$ . For a standard HMM, this transition probability is given by  $P(s_l|s_k) = \frac{\sum_{t=1}^T \xi_{kl}(t)}{\sum_{t=1}^T \gamma_k(t)}$ . However, at every  $t$  since we have  $N$  samples, we add an extra summation over all samples. Thus in our case the modified the transition probability is  $A(lk) = \frac{\sum_{t=1}^T \sum_{i=1}^N \xi_{kl}^i(t)}{\sum_{t=1}^T \sum_{i=1}^N \gamma_k(t)}$ . This is followed by normalizing every column  $A(k)$  such that it is a probability vector.

### 3.4 M step: Estimation of $\mu_i^j$ and $\sigma_i^j$

The Gaussian parameters  $\mu_i^j$  and  $\sigma_i^j$  are set to their maximum likelihood (ML) estimates for a mixture of Gaussians, [11] with a modification to incorporate the samples at every time-point. In the standard ML estimation of a mixture of  $n$  Gaussians, with  $M$  datapoints, with the  $j^{th}$  datapoint represented by  $d_j$ , the mean of the  $k^{th}$  Gaussian is given by  $\mu_k = \frac{\sum_{j=1}^M \delta_{kj}d_j}{\sum_{j=1}^M \delta_{kj}}$ . Here  $\delta_{ij}$  are the hidden variables, and represent the expectation that the  $j^{th}$  datapoint,  $d_j$ , is generated from the  $k^{th}$  Gaussian. In our case, we estimate Gaussians for every gene  $g_j$  in cells from every  $bs_k$ , resulting in  $mn$  Gaussians.  $x_t^i$ , i.e. the  $i^{th}$  forward sample at  $t$  represents the  $\delta$ . The datapoints from which these Gaussians are estimated are the expression vectors,  $y_1, \dots, y_T$ . A Gaussian for the  $j^{th}$  gene would use the  $y_t(j)$  components. In our model, ML estimates for  $\mu_k^j$  after incorporating  $N$  samples is given by  $\mu_k^j = \frac{\sum_{t=1}^T \sum_{i=1}^N x_t^i(k)y_t(j)}{\sum_{t=1}^T \sum_{i=1}^N x_t^i(k)}$ .  $\sigma_i^g$  is calculated using a similar modification to the standard ML formula, for the  $N$  samples.

## 4 Experiments

### 4.1 Yeast cell-cycle

We used our particle filter based approach to model the yeast cell-cycle process which controls cell growth and cell division. There are four primary phases in the cell-cycle: *Gap I (G1)*, *Synthesis (S)*, *Gap II (G2)* and *Mitosis (M)*. The yeast cell-cycle is an ordered set of events with *G1* being the first phase, followed by *S*, then *G2* and finally the last phase, *M*. The number of biological stages,  $n$ , correspond to the important events during the cell-cycle, which take place in a phase or during transitions between phases. We identified five stages ( $n = 5$ ) on the basis of [2] and [21]. These stages are :  $M$ ,  $M/G1$ ,  $G1$ ,  $S$ ,  $G2/M$ .  $M/G1$  is the transition between  $M$  and  $G1$  and  $G2/M$  is the transition between  $G2$  and  $M$ . We finally identified lists of genes for each of stages using existing biological knowledge of required genes for these stages [21, 2]. E.g. *CLN3* is required for early *G1* [15] and *SIC1* is required for the transition from  $M$  to  $G1$  [2]. These gene lists are used to perform biological validation of our results as described in Section 4.3.2.

### 4.2 Datasets

We tested our model for the cell-cycle on three datasets representing time-series microarray data from yeast cell-cycle experiments. These datasets are represented by the set  $Z = \{S_a, S_b, S_c\}$ .  $S_a$  and  $S_c$  were obtained from the cell-cycle experiment done by Spellman *et. al.* [21].  $S_a$  has  $T = 18$  timepoints and  $m = 712$  genes.  $S_c$  has  $T = 24$  timepoints and  $m = 710$  genes.  $L_a$  was obtained from Linda Breeden's cell-cycle experiment (unpublished).  $L_a$  has  $T = 13$  timepoints and  $n = 706$  genes. For all three datasets,  $n$  was set to 5, as described in section 4.1. Of these 712 genes, 696 genes were studied by [13]. The remaining 16 genes were added on the basis of biological literature [21, 2, 14, 15]. Two genes from  $S_c$  and six genes from  $L_a$  because were removed since their gene expressions were entirely absent in these datasets.  $\alpha$ -factor was used as the synchronizing technique for both  $S_a$  and  $L_a$  and *cdc15-2* was used for  $S_c$ .

### 4.3 Results and validation

The final outputs of our model are (i) a set of profiles,  $\{p_1, \dots, p_n\}$  and (ii) an assignment of member genes for every biological stage  $bs_k$ . A profile,  $p_k$  indicates the manner in which the population for  $k^{th}$  biological stage varies over the course of the microarray experiment. Thus  $p_k$  is a  $T$ -dimensional variable and  $p_k(t)$  indicates the fraction of cells in  $bs_k$  at  $t$ .  $p_k(t)$  is equal to  $\bar{x}_t(k)$ , where  $\bar{x}_t$  is the mean of  $D_t$ . To assign a gene  $g_j$  to a biological stage, we find the biological stage in which  $g_j$  has the highest mean expression i.e.  $\max(\mu_k^j)$ . This stage assignment for genes groups together similarly expressed genes thus providing a clustering mechanism for genes based on their activity levels in different biological stages.

#### 4.3.1 Statistical validation of results

To judge the statistical significance of our results, we tested our algorithm for consistency of the resultant profile sets and preservation of temporal dependencies. To verify that our algorithm converges to consistent profiles, we trained  $r$  different models for every dataset  $z$ ,  $z \in Z$ , resulting in  $r$  profile sets for every  $z$ . The average profile set and the standard deviation was then calculated from the  $r$  profile sets. Plots of the average profiles and standard deviations are shown in figures 1, 2, and 3. The standard deviations of the profiles indicate that the  $r$  profile sets, for a particular  $z$  were reasonably close. To test whether our approach preserves temporal dependencies, we used a randomized data confidence test similar to [6]. Let  $Q$  represent a model trained on data with temporal dependencies, i.e.  $z$ , and let  $R$  represent a model trained on randomly shuffled data  $z_s$ , where  $z_s$  is obtained by reordering the observation vectors in  $z$ . To compare  $Q$  and  $R$ , we calculated the log likelihood of the observed data  $z$ ,  $z \in Z$  for  $Q$  and  $R$ . The log likelihood,

$L$ , of the observed data, is  $P(y_1, y_2, \dots, y_T)$ . Using the chain rule,  $L = P(y_1) \prod_{t=2}^T P(y_t | y_1, \dots, y_{t-1})$ . Then using an approximation described in [5], the first  $t = 1$  term is  $P(y_1) = \sum_{i=1}^N P(y_1 | x_1) P(x_1^i)$  and the rest of the product is calculated as  $P(y_t | y_1, \dots, y_{t-1}) = \sum_{i=1}^N P(y_t | \hat{x}_t^i) \bar{w}_{t-1}^i$ . Let  $\rho_z$  and  $\tau_z$  denote the average likelihood and standard deviation from  $Q$  for a particular dataset  $z, z \in Z$ . Let  $\rho_{z_s}$  and  $\tau_{z_s}$  denote the average likelihood and standard deviation of  $z$  from  $R$ . These averages are obtained by starting with different initializations of the parameters for a model at the beginning of the EM training. We found that  $\rho_{z_s}$  was significantly smaller than  $\rho_z$ , indicating that our model is able to learn true temporal dependencies. The detailed results on each dataset is described in Table 1.

Dataset	True mean ( $\rho_z$ )	True stdev ( $\tau_z$ )	Shuffled mean ( $\rho_{z_s}$ )	Shuffled sdteve ( $\tau_{z_s}$ )
$S_a$	-738.7503	130.7760	-2612.045	286.0214
$S_b$	-6392.7	132.8753	-8371.642	332.229
$L_a$	2915.2406	23.2397	-1033.535	627.773

Table 1: Mean and standard deviations of model likelihood for the true and shuffled data

### 4.3.2 Biological validation of results

To show that the extracted profiles actually represent the stages of cell-cycle, we found a mapping between the profile set resulting from our model to the set  $\{M, M/G1, G1, S, G2/M\}$ . This was done by comparing the member genes assigned to  $bs_k$  with the list of genes known to be required for a biological stage. We found that in most cases, the member genes associated with a profile had a maximal overlap with only one set of known required genes. This allowed us to unambiguously associate an extracted profile with an element from  $\{M, M/G1, G1, S, G2/M\}$ . Moreover we found that each of the profiles had peaks at specific timepoints indicating that majority of the cells were in the corresponding biological stage. For  $S_a$ , Figure 2(d), and  $L_a$ , Figure 3(a), there was a profile with a sharp peak at  $t = 1$ . The most significant GO processes [10, 9] associated with these genes were related to  $\alpha$ -factor response. This is important, because  $\alpha$ -factor arrests cells somewhere in  $G1$ , and this profile indicates cells which are recovering from the affect of  $\alpha$ -factor and completing the rest of  $G1$ . We found similar behaviour in  $S_c$ , which used *cdc15-2* as the arresting mechanism.  $S_c$  had profile with a peak at  $t = 1$ , Figure 1(c), and the genes associated with this profile were involved in late  $M$  or  $M/G1$  phase. This was corroborated by the biological fact that *cdc15-2* arrests cells late in  $M$  phase. In addition to this, the ordering of the peaks of the profiles were in agreement with the stages in the cell-cycle, indicating the cyclic pattern of the cell-cycle.

## 5 Conclusion and future work

The primary goal of the approach described in this paper is to study the population dynamics of a temporal biological progress by solely making use of available microarray data. Our approach attempts to solve the CPD problem in a completely unsupervised fashion solely making use of the microarray data. The application of our approach to the yeast cell-cycle data shows that our model successfully recovers biologically meaningful profiles that demonstrate the temporal dynamics of the stages of the cell-cycle and provides an indication of the population synchrony over the microarray time course. Our method can also be employed as a clustering technique to group genes based on their activity levels in a biological stage. Finally, the randomization data confidence test proves that our approach incorporates and exploits temporal dependencies of the time-series data.

There are several directions in which this work can be extended. One direction is to investigate the possibility of finding a closed form solution to infer the cell fractions given the microarray data. Another

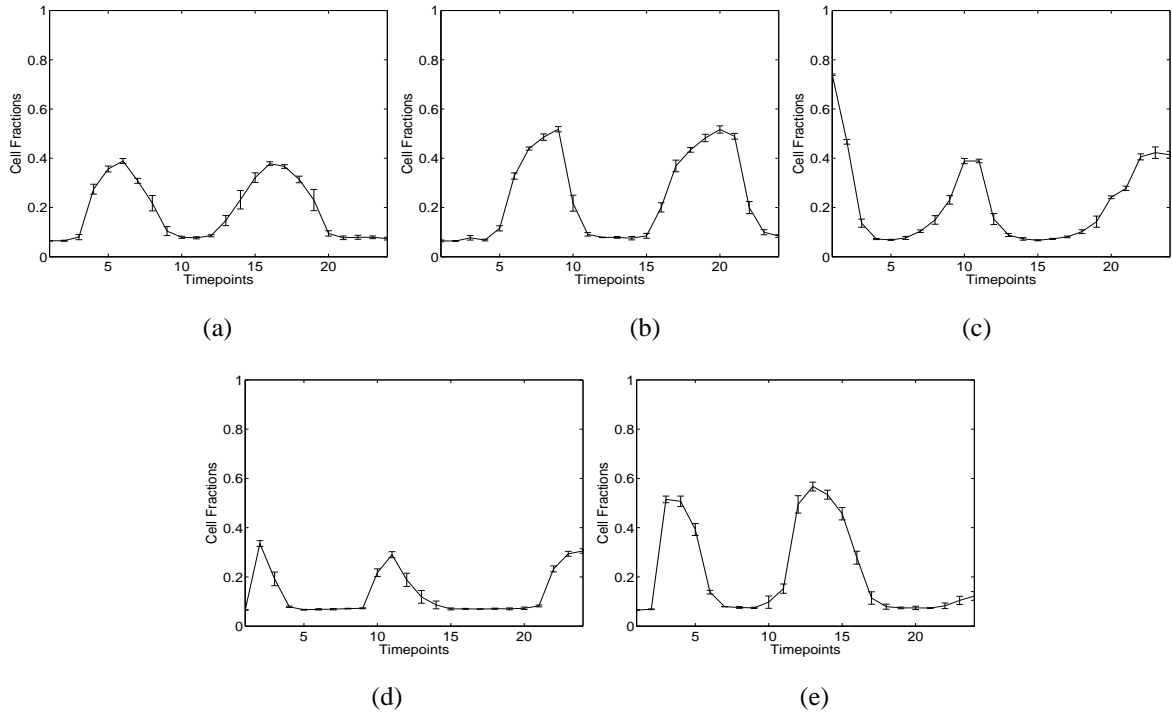


Figure 1: Profiles for  $S_c$  representing changing cell populations in each stage with time. (a) had maximal overlap with G2/M genes, (b) with M and M/G1 genes, (c) with M/G1 genes, (d) with G1 and S genes, (e) with S genes

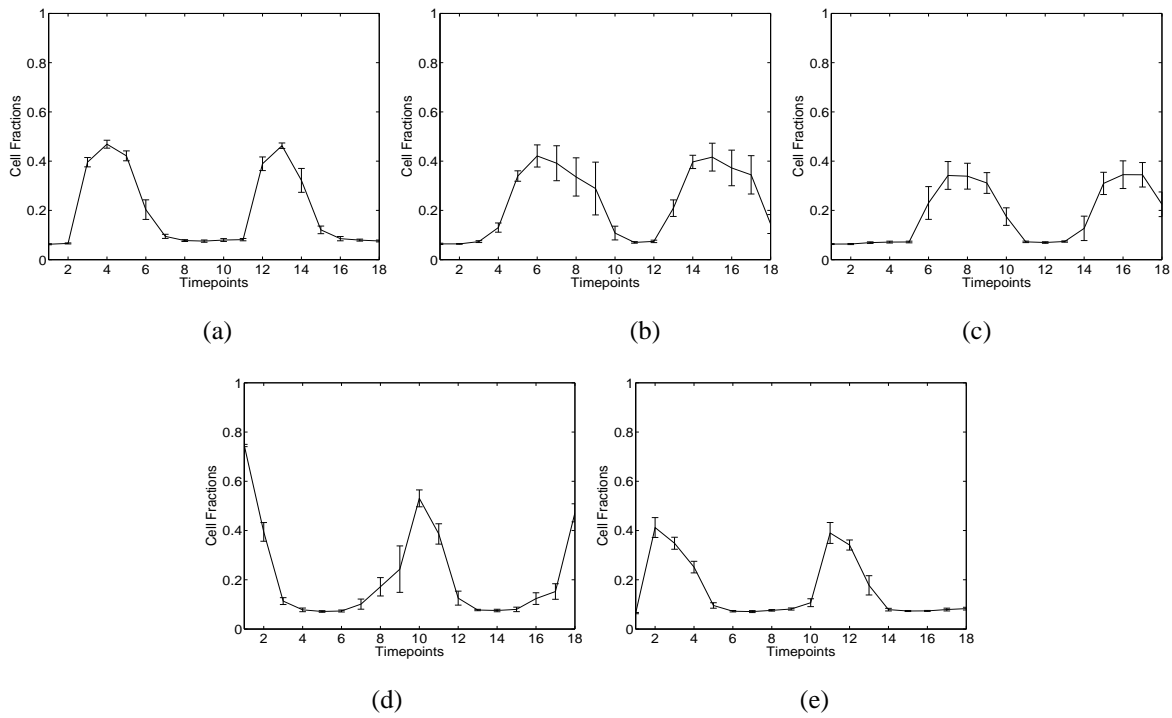


Figure 2: Profiles for  $S_a$  representing changing cell populations with time. (a) had maximal overlap with G1 genes, (b) with S genes, (c) with M and G2/M genes, (d) with  $\alpha$ -factor genes and early G1 genes, (e) with M/G1 genes

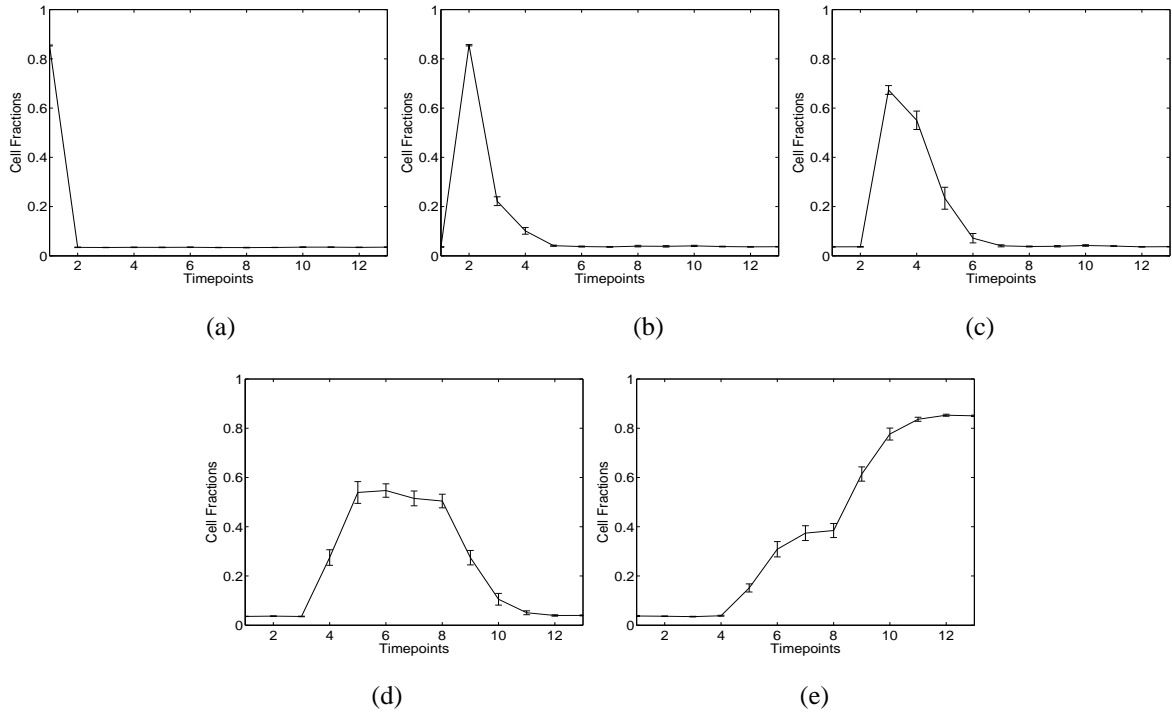


Figure 3: Profiles for  $L_a$  representing changing cell populations with time. (a) had maximal overlap with  $\alpha$ -factor genes and early G1 genes, (b) with M/G1 and G1 genes, (c) with S genes, (d) with G2/M, M and M/G1 genes. (e) had genes involved in cell growth and maintenance

direction is to find mathematical justifications for the modifications proposed in Section 2.7 and develop more sophisticated methods to resolve this problem. Finally our approach assumes that gene expressions at a timepoint are independent of each other given a biological stage. We hope to extend our model to incorporate dependencies between gene expressions.

## 6 Acknowledgments

We would like to thank Dr. Linda Breeden and Dr. Pramila Tata for making their dataset available to us and Dr. Peter Wentzell for helpful discussions. This work was supported by grants from the NSF (MCB-0092364) and NIH (HG02262).

## References

- [1] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 2002.
- [2] L. L. Breeden. Cyclin transcription: Timing is everything. *Current Biology*, 2000.
- [3] S. Cooper and K. Shedden. Microarray analysis of gene expression during the cell cycle. *Cell and Chromosome*, 2003.
- [4] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 1977.
- [5] A. Doucet, S. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering, 1998.
- [6] N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using bayesian networks to analyze expression data. In *RECOMB*, pages 127–135, 2000.
- [7] B. Futcher. Cell cycle synchronization. *Methods in Cell Science*, 1999.
- [8] N. Gordon, D. Salmond, and A. Smith. Novel approach to nonlinear /nongaussian bayesian state estimation. *IEE Proceedings-F (Radar and Signal Processing)*, 1993.
- [9] GO term finder. <http://db.yeastgenome.org/cgi-bin/go/gotermfinder>.
- [10] Gene Ontology. <http://www.geneontology.org>.
- [11] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2001.
- [12] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 1998.
- [13] P. Lu, A. Nakorchevskiy, and E. M. Marcotte. Expression deconvolution: A reinterpretation of DNA microarray data reveals dynamic changes in cell populations. *Proceedings of the National Academy of Sciences of the United States of America*, 2003.
- [14] V. L. Mackay, B. Mai, L. Waters, and L. L. Breeden. Early cell cycle box-mediated transcription of CLN3 and SWI4 contributes to the proper timing of the G1-to-S transition in budding yeast. *Molecular and Cellular Biology*, 2001.
- [15] B. Mai, S. Miles, and L. L. Breeden. Characterization of the ECB binding complex responsible for the M/G1-specific transcription of CLN3 and SWI4. *Molecular and Cellular Biology*, 2002.
- [16] T. Minka. Estimating a dirichlet distribution. Technical report, MIT, 2003.
- [17] A. Niemistö, T. Aho, H. Thesleff, M. Tiainen, K. Marjanen, M. Linne, and O. Yli-Harja. Estimation of population effects in synchronized budding yeast experiments. In *Image Processing: Algorithms and Systems II*, 2003.
- [18] A. Niemistö, M. Nykter, T. Aho, H. Jalovaara, K. Marjanen, M. Ahdesmäki, P. Ruusuvuori, M. Tiainen, M. Linne, and O. Yli-Harja. Distribution estimation of synchronized budding yeast population. In *Winter International Symposium on Information and Communication Technologies*, 2004.

- [19] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *IEEE*, 1989.
- [20] L. R. Rabiner and B. H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, 1986.
- [21] P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Andres, 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*, 1998.