Classification of Single-Cell Gene Expression Trajectories from Incomplete and Noisy Data

Alireza Karbalayghareh, Ulisses Braga-Neto, and Edward R. Dougherty

Abstract—This paper studies classification of gene-expression trajectories coming from two classes, healthy and mutated (cancerous) using Boolean networks with perturbation (BNps) to model the dynamics of each class at the state level. Each class has its own BNp, which is partially known based on gene pathways. We employ a Gaussian model at the observation level to show the expression values of the genes given the hidden binary states at each time point. We use expectation maximization (EM) to learn the BNps and the unknown model parameters, derive closed-form updates for the parameters, and propose a learning algorithm. After learning, a plug-in Bayes classifier is used to classify unlabeled trajectories, which can have missing data. Measuring gene expressions at different times yields trajectories only when measurements come from a single cell. In multiple-cell scenarios, the expression values are averages over many cells with possibly different states. Via the central-limit theorem, we propose another model for expression data in multiple-cell scenarios. Simulations demonstrate that single-cell trajectory data can outperform multiple-cell average expression data relative to classification error, especially in high-noise situations. We also consider data generated via a mammalian cell-cycle network, both the wild-type and with a common mutation affecting p27.

Index Terms—Gene regulatory network, probabilistic Boolean network, trajectory classification, Bayes classifier, expectation maximization, hidden Markov model, partially observed Boolean dynamical system, single-cell gene expression trajectory

1 Introduction

In a previous paper we have characterized the Bayes classifier and Bayes error for classification of steady-state trajectories observed in successive states in an original (wild-type) or mutated gene regulatory network (GRN) modeled via probabilistic Boolean networks (PBNs) [1]. In the present paper we consider classification when the networks are only partially known and the training data consist of labeled trajectories from an original and mutated network modeled as Boolean networks with perturbation (BNp), which is a special case of a PBN, observed indirectly through noise. The overall model is called a partially-observed Boolean dynamical system (POBDS) [2].

Owing to heterogeneity across samples and patients, it has long been recognized that it can be beneficial to use groups of genes as features. This can help avoid redundant information contained in selected genes, for instance, several genes in a pathway regulated by a single master gene [3]. The approach is to jointly analyze the expression levels of genes related by functionality, which can be obtained via transcriptome analysis [4], [5], [6], GO annotations [7], or other sources. Several methods have been proposed to measure the activity of a particular pathway: mean or median [8], first principle component [6], using a subset of genes in the pathway [9], and combining log-likelihood ratios of genes in the pathway [10] . Although these methods utilize multiple-gene features, they still rely on single measurements and do not take advantage of regulatory information in trajectory data.

Single-cell gene expression has recently become popular, as it is able to reveal the expressions of genes in many different cells in parallel in a single experiment, instead of bulk gene expression methods like conventional RNA-Seq in which the reported expression level of a gene is actually an average over cells with different states and possibly different types [11]. As a result, it has been utilized and proven to be a very effective alternative of bulk expression methods in various research studies. For instance, [12] demonstrated single-cell RNA-Seq (scRNA-Seq) as an effective strategy for classification of sensory neuron types. [13] used scRNA-Seq data to classify low quality cells. A massively parallel single-cell RNA profiling was used in [14] to classify retinal bipolar cells, where 15 previously known and two novel types were identified. Authors in [15] proposed a nonnegative matrix factorization (NMF) method as a robust unsupervised learning of cell subtypes from single-cell gene expression data. Zamanighomi et al. [16] proposed a method for unsupervised clustering of single-cell epigenetic data using single-cell ATAC-seq data.

Until recently, it has been difficult to obtain time-course expression data owing to the necessity of purifying and synchronizing a whole cell population [17]; however, new technologies are being developed for profiling single cells using single-cell RNA-Seq or quantitative PCR [18]. Individual cells can be captured via standard methods, such as flow cytometry, glass capillaries, or laser [19], and be measured

• *The authors are with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77843. E-mail: karbalayghareh@gmail.com, {ulisses, edward}@ece.tamu.edu.*

Manuscript received 17 Mar. 2017; revised 8 Aug. 2017; accepted 8 Oct. 2017. Date of publication 17 Oct. 2017; date of current version 4 Feb. 2019. (Corresponding author: Alireza Karbalayghareh.)
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TCBB.2017.2763946

at various time points. For instance, in [20], between 49 and 77 cells have been collected at each time for 4 total time points, and a software has been built to extract various gene-expression trajectories of individual cells. Results such as those obtained in the present paper, which demonstrate the clear advantage of single-cell trajectory data, will hopefully motivate the commercialization and extension of such technologies.

Single-cell expression measurements have enabled generating and using time-series data and discovering the regulatory information of genes, since bulk expression measurements, like RNA-Seq or microarrays, destroy crucial information by averaging signals from individual cells together [11]. However, lower amounts of mRNA in individual cells cause experimental issues which lead to dropout events [21], such that expressions of some genes are missed in some cells. Accordingly, in this paper, we also consider missing values of genes in order to better reflect the real data. Wang et al. [22] proposed a differential expression method using single-cell RNA-Seq time-series data for recovery of potential cell types from complex mixtures of multiple cell types. BNP-Seq, proposed in [23], is a Bayesian nonparametric differential expression analysis of count data, which might be beneficial if applied to single-cell RNA-Seq data to discover differentially expressed genes. Furthermore, [24] presented a probabilistic model with a Bayesian inference scheme to analyze single-cell time-series data, which was used for pseudotime estimation. Single-cell gene expression time-series measurements have also been employed to infer gene regulatory networks; for instance, single-cell expression measurements at four time points of blood development were used in [25] to synthesize a Boolean network model for 20 related transcription factors.

In this paper, gene regulation is modeled via BNps, in which states are binary vectors, and 1 and 0 represent On and Off, respectively (Binary representation is chosen because it models switch-like gene behavior and because it makes computation tractable, but the theory is directly extendable to any number of expression levels.) We consider a Gaussian observational model, in which the expression level of each gene given its state (hidden) follows a normal density with some unknown mean and variance. We observe the Gaussian expression values of n genes in m consecutive time points; however, to take account of missing data, at each time point there is a probability, p_{miss}, of not observing the expression of a gene. After observation of such trajectories, we estimate the unknown network parameters as well as the unknown network connections, which are partially known. For maximum likelihood estimation and inference, we use the Expectation Maximization (EM) approach to estimate the continuous parameters of the networks. We then plug in the estimated parameters and the inferred networks to the Bayes classifier. We study the effects of the different parameters on the average classification error over many random networks using trajectory data of different length and missing probability.

When gene-expression values are measured from tissues containing many cells, with genes not synchronized, a gene may be in different states at any time across the cell sample. Expression data derived from a multiple-cell scenario is approximated by average expression values across all states. To treat multiple-cell averaging, we consider averaged expression data and use a static model that does not take into account the dynamics of the networks. We compare the classification errors using trajectory data (single-cell) and averaged data (multiple-cell) in the simulation part and show that trajectories outperform averaged data if the trajectory length is sufficient, even with missing data.

The remainder of the paper is organized as follows. Section 2 provides an overview of a BNp and POBDS as the underlying model. Section 3 studies the classification of the single-cell trajectories of the gene expression data. Section 4 considers the case of the multiple-cell average expression data and its classification. Section 5 demonstrates the simulation results of each scenario, single-cell trajectories and multiple-cell averaging, for different sets of networks and parameters and also represents the comparisons between them. Finally, Section 6 concludes the paper. Note that preliminary results concerning some topics covered in the present paper were reported in [26].

2 Preliminaries

For a Boolean network (BN) on n genes, a truth table gives the functional relationships between the genes [27]. Each gene value $x_i \in \{0,1\}$, for $i = 1,\ldots,n$, at time $k+1$ is determined by the values of some predictor genes at time k via a Boolean function $f_i : \{0,1\}^n \rightarrow \{0,1\}$ in the truth table. In practice, f_i is a function of a small number of genes, K_i, called the *in-degree* of the gene x_i in the network. The in-degree of the network is the maximum of K_i's, that is, $K = \max_{i=1,\ldots,n} K_i$. A gene network can be represented as a graph with vertices representing genes and edges representing regulations. There is a state diagram of 2^n states corresponding to the truth table of the BN, representing the dynamics of the network. Given an initial state, a BN will eventually reach a set of states, called an *attractor cycle*, through which it will cycle endlessly. Each initial state corresponds to a unique attractor cycle, and the set of initial states leading to a specific attractor cycle is known as the *basin of attraction* (BOA) of the attractor cycle.

2.1 State Model

We allow stochasticity in our state model by using BNps instead of deterministic BNs. For BNps, perturbation is introduced with a probability p by which the state of the network can be randomly changed at any time. Implicitly, we assume that there is an independent identically distributed (i.i.d.) random perturbation vector at each time k, denoted by $\mathbf{n}_k \in \{0,1\}^n$, such that the ith gene flips at time k if the i th component of $\mathbf{n}_k$ is equal to 1. Therefore, the dynamical model of the states can be expressed as

$$\mathbf{X}_{k+1} = \mathbf{f}(\mathbf{X}_k) \oplus \mathbf{n}_{k+1}, \quad k = 0,1,2,\ldots, \tag{1}$$

where $\mathbf{X}_k = [x_1(k), x_2(k), \ldots, x_n(k)]^T$ is a binary state vector, called a *gene activity profile* (GAP), at time k, in which $x_i(k)$ indicates the expression level of the ith gene at time k (either 0 or 1); $\mathbf{f} = [f_1, f_2, \ldots, f_n]^T : \{0,1\}^n \rightarrow \{0,1\}^n$ is the vector of the network functions, in which f_i shows the expression level of the ith gene at time $k+1$ when the system lies in the state $\mathbf{X}_k$ at time k; $\mathbf{n}_k = [n_1(k), n_2(k), \ldots, n_n(k)]^T$ is the perturbation vector at time k, in which $n_1(k), n_2(k), \ldots,$

$n_n(k)$ are i.i.d. Bernoulli random variables for every k with the parameter $p = P(n_i(k) = 1)$ for every $i = 1, \ldots, n$; and $\oplus$ is component-wise modulo 2 addition.

The existence of perturbation makes the corresponding Markov chain of a BNp irreducible. Hence, the network possesses a steady-state distribution π describing its long-run behavior. A BNp inherits the attractor structure from the original BN without perturbation, the difference being that a random perturbation can cause a BNp to jump out of an attractor cycle, perhaps then transitioning to a different attractor cycle. If p is sufficiently small, π will reflect the attractor structure within the original network. We can derive the transition probability matrix (TPM) if we know the truth table and the perturbation probability of a BNp. As a result, the steady-state distribution π can be computed as well.

We assume that the networks are partially known, perhaps from biological pathway knowledge or previous partial inference, and the missing model parameters are estimated from the new trajectory data. One could, in principle, assume that nothing is known about the BNs except the genes and depend entirely on the data, but we are generally interested in using prior knowledge to facilitate classifier design.

Since we do a supervised classification between two classes, healthy and a specific mutated phenotype, the mutated genes of that mutated phenotype determine the desired BN. For example, in the paper we have considered a specific phenotype in which the gene p27 is mutated. As a result, we are required to use a pathway and a BN which involves the gene p27; one such BN is the known cell-cycle BN which we have utilized as our healthy class and its mutated version of knocked out p27 as the mutated class. In general, if we are interested in a supervised classification between a healthy class and a mutated class in which some genes are mutated, we need to use a BN which involves those mutated genes.

2.2 Observation Model

Our model for gene expression is the partially-observed Boolean dynamical system [2], which is a special case of a hidden Markov model (HMM). Having defined the state transition model as a BNp, we now define the observation model given the hidden states by assuming that the expression level of each gene at any time comes from a Gaussian distribution whose mean value is specified by that gene's binary state value, which is hidden. In other words, depending on whether a gene is active or not, its expression value comes from two Gaussian distributions with two different means. The observation model for the jth gene at time k is

$$p\big(y_j(k)|x_j(k)\big) \sim \mathcal{N}\big(\lambda + \delta_j x_j(k), \sigma^2\big), \quad j = 1, 2, \ldots, n, \quad (2)$$

where $x_j(k)$ is the hidden binary state (0 or 1) of the jth gene at time k, and $y_j(k)$ is the observed expression value of the jth gene at time k. The variance σ^2 is constant, but the mean varies over time, as the value of $x_j(k)$ is changing according to the state dynamics (1). This shows that when the jth gene is off (suppressed) and on (expressed), its observed expression values come from Gaussian distributions with the means of λ and $\lambda + \delta_j$, respectively, and with the same variance σ^2. In (2), λ is the baseline expression level of the genes, which depends on the sequencing technology, and δ_j is the activation coefficient of the jth gene, which determines the level of the expression for the jth gene when it is on. Although we can proceed with arbitrary values of δ_j for different genes, for the sake of simplicity we assume the same activation coefficient for all the genes, that is, $\delta_j = \delta$ for $j = 1, \ldots, n$.

We denote the expression values of all n genes at time k by the vector $\mathbf{Y}_k = [y_1(k), \ldots, y_n(k)]^T$. If we assume that, at any time point k, the expression value of each gene given its binary state is independent of the expressions of other genes given their corresponding binary states, we can write,

$$\mathbf{Y}_k = \lambda \mathbf{1}_n + \delta \mathbf{X}_k + \epsilon, \quad k = 1, 2, \ldots, \quad (3)$$

where $\mathbf{1}_n$ is an $n \times 1$ all-one vector and $\epsilon \sim \mathcal{N}(0, \sigma^2 I_n)$ is a $n \times 1$ multivariate Gaussian random variable of zero mean and diagonal covariance matrix (I_n is $n \times n$ identity matrix) showing the variability across the samples. The state vector $\mathbf{X}_k$ in (3) is hidden (not observed), and the conditional distribution of $\mathbf{Y}_k$ given $\mathbf{X}_k$ is

$$p(\mathbf{Y}_k|\mathbf{X}_k) \sim \mathcal{N}(\lambda \mathbf{1}_n + \delta \mathbf{X}_k, \sigma^2 I_n), \quad k = 1, 2, \ldots. \quad (4)$$

Note that there are two types of variability: intra-subject and inter-subject. Intra-subject variability is sometimes called within-subject variability and refers to the variability of the samples in one subject, for example, the variability seen in the expression values of the genes in one individual at different times. Subject in this context means either cell, organism, or individual. Inter-subject variability is sometimes called between-subject variability and represents the variability among the samples of different subjects. For example, the variability seen in the expression values of the genes in different individuals refers to inter-subject variability. For avoiding confusion, we use the term "inter-cell variability" to refer to the variability across different cells of an individual, which will be used in Section 4 for the analysis of the multiple-cell scenario. In this section, we do single-cell analysis and do not deal with inter-cell variability, since each individual has a single cell to be used for expression measurements. We should note that in (3), ϵ accounts for both the inter- and intra-subject variability.

3 Classification of Trajectories with Missing Data in Single-Cell Scenarios

Assume there are two BNps corresponding to the healthy and mutated (cancerous) classes, each having n genes, and we partially know the networks but do not know the model parameters p, λ, δ, and σ^2. The healthy and mutated networks may have distinct model parameter values. Using D observed trajectories, $\mathbb{Y} = \{\mathcal{Y}^{(1)}, \mathcal{Y}^{(2)}, \ldots, \mathcal{Y}^{(D)}\}$, we infer the unknown parameters and connections of each network. $\mathbb{Y}$ may be incomplete, meaning that there may be missing data. Without missing data, each trajectory $\mathcal{Y}^{(d)}$, for $d = 1, \ldots, D$, has the expression values of the n genes in m consecutive time points. However, if each gene at each time point has the probability p_{miss} of being missed, then each observed trajectory has the form $\mathcal{Y}^{(d)} = [\mathbf{Y}^{(d)}_{i_1}, \ldots, \mathbf{Y}^{(d)}_{i_{m(d)}}]$, where $T^{(d)}_{obs} = \{i_1, \ldots, i_{m(d)}\}$ is the set of time points at which at least one gene is observed.

For the maximum likelihood (ML) problem, the search space consists of both discrete and continuous parts. The space of network functions is discrete and that of the parameters is continuous. Suppose $\mathbf{F} = \{\mathbf{f}^1, \mathbf{f}^2, \ldots, \mathbf{f}^M\}$ is the uncertainty set of M network functions containing the unknown true network function in (1). We wish to infer the true network function using the observation data. In each class, healthy or mutated, we assume an uncertainty set of network functions $\mathbf{F}$. Although there are many biologically confirmed gene pathways from which Boolean networks can be constructed, we are likely to be uncertain about some regulations and interactions between some genes. In such cases, we can form an uncertainty class $\mathbf{F}$ of network functions, each being a possible network function which would be inferred from the observed data. For instance, assume we know that the gene A regulates the gene B but are not sure about the type of regulation, that is, activator or suppressor. As such, in our uncertainty class of network functions we let $\mathbf{f}^1$ and $\mathbf{f}^2$ be the network functions for the cases that regulator A to B is activator and suppressor, respectively. In a similar way, we can consider any kind of uncertainty in the network structure, and the true network is inferred from the observed trajectories.

Suppose the model parameters are defined as the vector $\theta = [p, \lambda, \delta, \sigma^2]^T$. For any given network function $\mathbf{f}^i$, $i = 1, \ldots, M$, we employ the EM algorithm to find the optimal parameters θ by

$$\hat{\theta}_i = \underset{\theta}{\operatorname{argmax}}\, p(\mathbb{Y}|\mathbf{f}^i, \theta), \tag{5}$$

where $p(\mathbb{Y}|\mathbf{f}^i, \theta)$ is the likelihood of the observation trajectory set $\mathbb{Y}$ given that the network function is $\mathbf{f}^i$ and the parameter is θ. The ML inferred network function and estimated parameters are then derived as

$$(\hat{\mathbf{f}}, \hat{\theta}) = \arg \max_{(\mathbf{f},\theta)\in\{(\mathbf{f}^1,\hat{\theta}_1),\ldots,(\mathbf{f}^M,\hat{\theta}_M)\}} p(\mathbb{Y}|\mathbf{f}, \theta). \tag{6}$$

3.1 EM Algorithm for Finding θ

In (5), the network function is given, and we are supposed to find the ML estimation for θ. To ease notation, suppose the network function in (5) is denoted by $\mathbf{f}$. As there are hidden states in the model, we employ the EM algorithm to estimate the parameters. The EM algorithm can be described simply as repeating the following steps until convergence:

1- E-step: $Q(\theta, \theta^{(s)}) = \sum_{\mathbb{X}} \log [p(\mathbb{X}, \mathbb{Y}|\theta)] P(\mathbb{X}|\mathbb{Y}, \theta^{(s)})$,

2- M-step: $\theta^{(s+1)} = \operatorname{argmax}_\theta Q(\theta, \theta^{(s)})$,

where $\mathbb{X} = \{\mathcal{X}^{(1)}, \ldots, \mathcal{X}^{(D)}\}$ contains the hidden state trajectories corresponding to D observed trajectories, such that $\mathcal{X}^{(d)} = [\mathbf{X}_1^{(d)}, \ldots, \mathbf{X}_m^{(d)}]$ are the hidden states of the dth trajectory from time 1 to m.

3.1.1 E-Step

Since the D trajectory observations are i.i.d., we can write the joint log-likelihood of $\mathbb{X}$ and $\mathbb{Y}$ as

$$\log [p(\mathbb{X}, \mathbb{Y}|\theta)] = \sum_{d=1}^{D} \log [p(\mathcal{X}^{(d)}, \mathcal{Y}^{(d)}|\theta)]. \tag{7}$$

The joint likelihood of each dth observed trajectory $\mathcal{Y}^{(d)}$ and its corresponding hidden trajectory $\mathcal{X}^{(d)}$ can be factored as

$$p(\mathcal{X}^{(d)}, \mathcal{Y}^{(d)}|\theta) = P(\mathbf{X}_1^{(d)}) \prod_{k=1}^{m-1} P(\mathbf{X}_{k+1}^{(d)}|\mathbf{X}_k^{(d)}) \prod_{k\in T_{obs}^{(d)}} p(\mathbf{Y}_k^{(d)}|\mathbf{X}_k^{(d)}), \tag{8}$$

where, under the usual biological assumption of steady-state observations, $P(\mathbf{X}_1^{(d)})$ is the steady-state probability of state $\mathbf{X}_1^{(d)}$. Let $\mathbf{x}^i$ denote the $n \times 1$ binary vector of the state i, for $i = 1, \ldots, 2^n$. For example, if $n = 4$, then $\mathbf{x}^1 = [0, 0, 0, 0]^T$ and $\mathbf{x}^{10} = [1, 0, 1, 1]^T$. Denote the steady-state distribution by the 1×2^n vector $\pi = [\pi_1, \ldots, \pi_{2^n}]$, where π_i is the steady-state probability of being in the ith state. Then $P(\mathbf{X}_1^{(d)} = \mathbf{x}^i) = \pi_i$ for any $d = 1, \ldots, D$ and $i = 1, \ldots, 2^n$. Should we drop the steady-state assumption, then $P(\mathbf{X}_1^{(d)})$ is an arbitrary distribution to be estimated from the observed data. The second term in (8) is the probability of transitioning from state $\mathbf{X}_k^{(d)}$ at time k to state $\mathbf{X}_{k+1}^{(d)}$ at time $k+1$, which using (1) can be written as

$$P(\mathbf{X}_{k+1}^{(d)}|\mathbf{X}_k^{(d)}) = p^{\mathbf{d}(\mathbf{X}_{k+1}^{(d)}, \mathbf{f}(\mathbf{X}_k^{(d)}))} (1-p)^{n-\mathbf{d}(\mathbf{X}_{k+1}^{(d)}, \mathbf{f}(\mathbf{X}_k^{(d)}))}, \tag{9}$$

where $\mathbf{d}(\mathbf{X}_{k+1}^{(d)}, \mathbf{f}(\mathbf{X}_k^{(d)}))$ denotes the Hamming distance between the two binary vectors $\mathbf{X}_{k+1}^{(d)}$ and $\mathbf{f}(\mathbf{X}_k^{(d)})$. The probabilities in (9) are the entries in the transition probability matrix. The third term in (8), $p(\mathbf{Y}_k^{(d)}|\mathbf{X}_k^{(d)})$, is the likelihood of the gene-expression vector $\mathbf{Y}_k^{(d)}$ at time k given its corresponding hidden state vector $\mathbf{X}_k^{(d)}$. In the absence of missing data, $\mathbf{Y}_k^{(d)}$ contains the expression values of all n genes, but with missing data some expression values may not appear in $\mathbf{Y}_k^{(d)}$. Let $G_k^{(d)}$ denote the set of genes whose expressions have been observed at time k of the d th trajectory. Then from (4), we have,

$$p(\mathbf{Y}_k^{(d)}|\mathbf{X}_k^{(d)}) = \prod_{j\in G_k^{(d)}} (2\pi\sigma^2)^{-\frac{1}{2}} \exp\left[\frac{-\left(y_j^{(d)}(k) - \lambda - \delta x_j^{(d)}(k)\right)^2}{2\sigma^2}\right]. \tag{10}$$

Using (7) (8) (9) and (10)–(10), the joint log-likelihood can be written as in (11).

$$\begin{aligned}\log [p(\mathbb{X}, \mathbb{Y}|\theta)] = \sum_{d=1}^{D} \Bigg\{ &\log P(\mathbf{X}_1^{(d)}) + \sum_{k=1}^{m-1} \Big[\mathbf{d}(\mathbf{X}_{k+1}^{(d)}, \mathbf{f}(\mathbf{X}_k^{(d)})) \log p \\ &+ [n - \mathbf{d}(\mathbf{X}_{k+1}^{(d)}, \mathbf{f}(\mathbf{X}_k^{(d)}))] \log (1-p)\Big] \\ &+ \sum_{k\in T_{obs}^{(d)}} \sum_{j\in G_k^{(d)}} \left[-\frac{1}{2}\log 2\pi\sigma^2 - \frac{\left(y_j^{(d)}(k) - \lambda - \delta x_j^{(d)}(k)\right)^2}{2\sigma^2} \right] \Bigg\}.\end{aligned} \tag{11}$$

Now we can compute $Q(\theta, \theta^s)$. After some straightforward simplifications and dropping the constant parts, $Q(\theta, \theta^s)$ can be derived as in (12),

$$Q(\theta,\theta^{(s)}) = \sum_{d=1}^{D}\sum_{i=1}^{2^n}\log(\pi_i)\Pi_i^{(d,s)}(1) + \sum_{d=1}^{D}\sum_{k=1}^{m-1}\sum_{i=1}^{2^n}\sum_{j=1}^{2^n} \Big[\mathbf{d}(\mathbf{x}^j,\mathbf{f}(\mathbf{x}^i))\log p + [n-\mathbf{d}(\mathbf{x}^j,\mathbf{f}(\mathbf{x}^i))]\log(1-p)\Big]\Xi_{i,j}^{(d,s)}(k) + \sum_{d=1}^{D}\sum_{k\in T_{obs}^{(d)}}\sum_{i=1}^{2^n}\sum_{j\in G_k^{(d)}}\left[-\frac{1}{2}\log\sigma^2 - \frac{\left(y_j^{(d)}(k)-\lambda-\delta\mathbf{x}_j^i\right)^2}{2\sigma^2}\right]\Pi_i^{(d,s)}(k). \tag{12}$$

where,

$$\Pi_i^{(d,s)}(k) = P(\mathbf{X}_k^{(d)} = \mathbf{x}^i | \mathcal{Y}^{(d)}, \theta^{(s)}), \tag{13}$$

for any $i = 1,\dots,2^n$, $k = 1,\dots,m$, and $d = 1,\dots,D$, is the posterior probability of the state i at time k after the observation of the dth trajectory, and given the parameter vector $\theta^{(s)}$. Furthermore, in (12),

$$\Xi_{i,j}^{(d,s)}(k) = P(\mathbf{X}_k^{(d)} = \mathbf{x}^i, \mathbf{X}_{k+1}^{(d)} = \mathbf{x}^j | \mathcal{Y}^{(d)}, \theta^{(s)}), \tag{14}$$

for any $i,j = 1,\dots,2^n$, $k = 1,\dots,m-1$, and $d = 1,\dots,D$, is the posterior probability of two consecutive states being i and j, respectively, at times k and $k+1$ after the observation of the dth trajectory and given the parameter vector $\theta^{(s)}$.

3.1.2 M-Step

Having derived $Q(\theta,\theta^{(s)})$, we address the second step of the EM method, which is the maximization of $Q(\theta,\theta^{(s)})$. We take the derivative of $Q(\theta,\theta^{(s)})$ with respect to θ. The derivatives of $Q(\theta,\theta^{(s)})$ with respect to p, λ, δ, and σ^2 are

$$\frac{\partial Q}{\partial p} = \sum_{d=1}^{D}\sum_{\mathbf{i}=1}^{2^n}\frac{\pi_i'}{\pi_i}\Pi_i^{(d,s)}(1) + \sum_{d=1}^{D}\sum_{k=1}^{m-1}\sum_{i=1}^{2^n}\sum_{j=1}^{2^n}\left[\frac{\mathbf{d}(\mathbf{x}^j,\mathbf{f}(\mathbf{x}^i))}{p(1-p)} - \frac{n}{1-p}\right]\Xi_{i,j}^{(d,s)}(k), \tag{15}$$

$$\frac{\partial Q}{\partial \lambda} = \frac{1}{\sigma^2}\sum_{d=1}^{D}\sum_{k\in T_{obs}^{(d)}}\sum_{i=1}^{2^n}\sum_{j\in G_k^{(d)}}\left(y_j^{(d)}(k)-\lambda-\delta\mathbf{x}_j^i\right)\Pi_i^{(d,s)}(k), \tag{16}$$

$$\frac{\partial Q}{\partial \delta} = \frac{1}{\sigma^2}\sum_{d=1}^{D}\sum_{k\in T_{obs}^{(d)}}\sum_{i=1}^{2^n}\sum_{j\in G_k^{(d)}}\mathbf{x}_j^i\left(y_j^{(d)}(k)-\lambda-\delta\mathbf{x}_j^i\right)\Pi_i^{(d,s)}(k), \tag{17}$$

$$\frac{\partial Q}{\partial \sigma^2} = \frac{-1}{2\sigma^2}\sum_{d=1}^{D}\sum_{k\in T_{obs}^{(d)}}\sum_{i=1}^{2^n}\sum_{j\in G_k^{(d)}}\left[1-\frac{\left(y_j^{(d)}(k)-\lambda-\delta\mathbf{x}_j^i\right)^2}{\sigma^2}\right]\Pi_i^{(d,s)}(k), \tag{18}$$

respectively. π_i' in (15) is the derivative of the steady-state distribution of the state i with respect to p, that is, $\pi_i' = \frac{\partial \pi_i}{\partial p}$. To find π', we start with the fact that the steady-state distribution $\pi = [\pi_1, \pi_2, \dots, \pi_{2^n}]$ satisfies

$$\pi = \pi A, \tag{19}$$

$$\sum_{i=1}^{2^n}\pi_i = 1, \tag{20}$$

where A is the TPM with the corresponding entries (9),

$$A_{i,j} = p^{\mathbf{d}(\mathbf{x}^j,\mathbf{f}(\mathbf{x}^i))}(1-p)^{n-\mathbf{d}(\mathbf{x}^j,\mathbf{f}(\mathbf{x}^i))}. \tag{21}$$

Taking the derivative of both sides in (19) and (20) with respect to p yields

$$\pi'(I-A) = \pi A', \tag{22}$$

$$\sum_{i=1}^{2^n}\pi_i' = 0, \tag{23}$$

where A' is the derivative of the TPM with respect to p and, using (21), can be written in terms of A as

$$A_{i,j}' = \left(\frac{\mathbf{d}(\mathbf{x}^j,\mathbf{f}(\mathbf{x}^i)) - np}{p(1-p)}\right)A_{i,j}. \tag{24}$$

π' is easily found from the linear Equations (22) and (23).

Given the derivatives in (16), (17), and (18), thanks to the specific form of the Gaussian distribution, we can derive closed-formed solutions for λ, δ, and σ^2 by setting the derivatives equal to zero. Such closed-form solutions considerably reduce the complexity of the EM algorithm because they eliminate the iterative computations required for the algorithms like gradient descent in every M-step. Define ρ_1 and ρ_2 by

$$\rho_1 = \sum_{d=1}^{D}\sum_{k\in T_{obs}^{(d)}}\sum_{i=1}^{2^n}\sum_{j\in G_k^{(d)}} y_j^{(d)}(k)\Pi_i^{(d,s)}(k) = \sum_{d=1}^{D}\sum_{k\in T_{obs}^{(d)}}\sum_{j\in G_k^{(d)}} y_j^{(d)}(k), \tag{25}$$

$$\rho_2 = \sum_{d=1}^{D}\sum_{k\in T_{obs}^{(d)}}\sum_{i=1}^{2^n}\sum_{j\in G_k^{(d)}}\Pi_i^{(d,s)}(k) = \sum_{d=1}^{D}\sum_{k\in T_{obs}^{(d)}}\sum_{j\in G_k^{(d)}} 1, \tag{26}$$

where we have used the fact that the summation of the posterior probabilities of the states is one, $\sum_{i=1}^{2^n}\Pi_i^{d,s}(k) = 1$, for any d and k and s. Define ρ_3 and ρ_4 by

$$\rho_3 = \sum_{d=1}^{D}\sum_{k\in T_{obs}^{(d)}}\sum_{i=1}^{2^n}\sum_{j\in G_k^{(d)}}\mathbf{x}_j^i\Pi_i^{(d,s)}(k), \tag{27}$$

$$\rho_4 = \sum_{d=1}^{D}\sum_{k\in T_{obs}^{(d)}}\sum_{i=1}^{2^n}\sum_{j\in G_k^{(d)}}\mathbf{x}_j^i y_j^{(d)}(k)\Pi_i^{(d,s)}(k). \tag{28}$$

Setting the derivatives in (16) and (17) equal to zero yields two linear equations for λ and δ:

$$\rho_2\lambda + \rho_3\delta = \rho_1,\ \ \rho_3\lambda + \rho_3\delta = \rho_4. \tag{29}$$

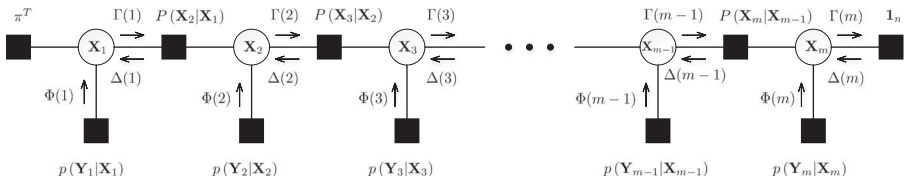

Fig. 1. Factor graph of the HMM.

If $\rho_3 \neq 0$, then

$$\lambda^{(s+1)} = \frac{\rho_1 - \rho_4}{\rho_2 - \rho_3}, \tag{30}$$

$$\delta^{(s+1)} = \frac{\rho_2\rho_4 - \rho_1\rho_3}{\rho_2\rho_3 - \rho_3^2}. \tag{31}$$

If $\rho_3 = 0$, then $\rho_4 = 0$, $\lambda^{(s+1)} = \frac{\rho_1}{\rho_2}$, but δ cannot be found. In this case, we define $\delta^{(s+1)} = \delta^{(s)}$. Now define ρ_5 by

$$\rho_5 = \sum_{d=1}^{D} \sum_{k \in T_{obs}^{(d)}} \sum_{i=1}^{2^n} \sum_{j \in G_k^{(d)}} \left(y_j^{(d)}(k) - \lambda^{(s+1)} - \delta^{(s+1)}\mathbf{x}_j^i\right)^2 \Pi_i^{(d,s)}(k). \tag{32}$$

Setting the derivative in (18) equal to zero gives the following solution for σ^2:

$$\sigma^{2(s+1)} = \frac{\rho_5}{\rho_2}. \tag{33}$$

To find a closed-form solution for p note that the derivative of Q with respect to p in (15) consists of two terms. In the first term, π and π' are not explicit functions of p, which means that we are unable to derive a closed-form solution for p by setting the derivative equal to zero. From simulations, we found that the second term in (15) plays a much more important role than the first term, that is, has a much larger value. Hence, a good approximation results from omitting the first term in (15) and setting the second to zero, which gives the following approximate closed-form solution for p (we tested its accuracy and it can correctly estimate the real p)

$$p^{(s+1)} = \frac{\rho_6}{nD(m-1)}, \tag{34}$$

where ρ_6 is defined as

$$\rho_6 = \sum_{d=1}^{D} \sum_{k=1}^{m-1} \sum_{i=1}^{2^n} \sum_{j=1}^{2^n} \mathbf{d}(\mathbf{x}^j, \mathbf{f}(\mathbf{x}^i)) \Xi_{i,j}^{(d,s)}(k). \tag{35}$$

Note that in deriving (34) we have used the fact that $\sum_{i=1}^{2^n}\sum_{j=1}^{2^n} \Xi_{i,j}^{(d,s)}(k) = 1$ for any k, d, and s, since $\Xi_{i,j}^{(d,s)}(k)$ is the posterior probability of two consecutive states being i and j at times k and $k+1$, respectively.

3.1.3 Computing the Posterior Probabilities of the States

As our model is an HMM, the posterior probabilities in (13) and (14) can be efficiently computed using the forward-backward algorithm, whose complexity is linear in m. From Bayes rule, we know that the posterior probabilities of states such as those in (13) and (14) can be computed using the joint distribution of the states $\mathcal{X}$ and observation trajectories $\mathcal{Y}$, which are factored as in (8). A joint distribution (8) for an HMM can be represented by a factor graph as in Fig. 1 [28]. In this figure, the circles show the state variable from time 1 to m; the factor nodes (black) between the state nodes denote the transition probabilities between two consecutive states; the factor nodes under the state nodes represent the likelihood of each observation given its corresponding state. The rightmost factor node is an all-one vector and the left-most factor node is the initial distribution of the states, which based on our assumption is the steady-state distribution. The message-passing is done through the graph by defining so-called forward, Γ, and backward parameters, Δ . It can be shown that the posteriors in (13) and (14) can be derived as [28]

$$\Pi_i^{(d,s)}(k) = \frac{\Gamma_i^{(d,s)}(k)\Delta_i^{(d,s)}(k)}{\sum_{r=1}^{2^n}\Gamma_r^{(d,s)}(k)\Delta_r^{(d,s)}(k)}, \tag{36}$$

$$\Xi_{i,j}^{(d,s)}(k) = \frac{\Gamma_i^{(d,s)}(k)A_{i,j}^{(s)}\Delta_j^{(d,s)}(k+1)\Phi_j^{(d,s)}(k+1)}{\sum_{r=1}^{2^n}\Gamma_r^{(d,s)}(m)}, \tag{37}$$

where $A_{i,j}^{(s)} = P(\mathbf{X}_{k+1} = \mathbf{x}^j | \mathbf{X}_k = \mathbf{x}^i, \theta = \theta^{(s)})$ is the transition matrix defined in (21), and $\Phi^{(d,s)}(k)$ is a $2^n \times 1$ vector at time k, whose jth entry is defined as $\Phi_j^{(d,s)}(k) = p(\mathbf{Y}_k^{(d)} | \mathbf{X}_k^{(d)} = \mathbf{x}^j, \theta = \theta^{(s)})$, which can be computed using (10). Furthermore, $\Gamma_i^{(d,s)}(k)$ and $\Delta_i^{(d,s)}(k)$ are respectively the forward and backward parameters, defined and recursively computed by

$$\begin{aligned} \Gamma_i^{(d,s)}(k) &= p(\mathbf{Y}_1^{(d)}, \ldots, \mathbf{Y}_k^{(d)}, \mathbf{X}_k^{(d)} = \mathbf{x}^i | \theta^{(s)}), \\ \Gamma_i^{(d,s)}(1) &= \pi_i^{(s)}\Phi_i^{(d,s)}(1), \\ \Gamma_j^{(d,s)}(k+1) &= \Phi_j^{(d,s)}(k+1)\sum_{i=1}^{2^n}\Gamma_i^{(d,s)}(k)A_{i,j}^{(s)}, \end{aligned} \tag{38}$$

and

$$\begin{aligned} \Delta_i^{(d,s)}(k) &= p(\mathbf{Y}_{k+1}^{(d)}, \ldots, \mathbf{Y}_m^{(d)} | \mathbf{X}_k^{(d)} = \mathbf{x}^i, \theta^{(s)}), \\ \Delta_i^{(d,s)}(m) &= 1, \\ \Delta_i^{(d,s)}(k) &= \sum_{j=1}^{2^n}\Delta_j^{(d,s)}(k+1)A_{i,j}^{(s)}\Phi_j^{(d,s)}(k+1), \end{aligned} \tag{39}$$

for any $k = 1, \ldots, m-1$.

Define the vectors $\Gamma^{(d,s)}(k) = [\Gamma_1^{(d,s)}(k), \ldots, \Gamma_{2^n}^{(d,s)}(k)]^T$ and $\Delta^{(d,s)}(k) = [\Delta_1^{(d,s)}(k), \ldots, \Delta_{2^n}^{(d,s)}(k)]^T$. From (38) and (39), we have the following recursions in vector-matrix form

$$\begin{aligned} \Gamma^{(d,s)}(1) &= {\pi^{(s)}}^T \circ \Phi^{(d,s)}(1), \\ \Gamma^{(d,s)}(k+1) &= \left[A^{(s)^T}\Gamma^{(d,s)}(k)\right] \circ \Phi^{(d,s)}(k+1), \end{aligned} \tag{40}$$

and

$$\begin{aligned}\Delta^{(d,s)}(m) &= \mathbf{1}_{2^n},\\ \Delta^{(d,s)}(k) &= A^{(s)}\Big[\Delta^{(d,s)}(k+1)\circ\Phi^{(d,s)}(k+1)\Big],\end{aligned} \tag{41}$$

where $\mathbf{1}_{2^n}$ is the all-one column vector of length 2^n and $\circ$ denotes the Hadamard product (or component-wise product). The superscript T denotes transpose. Now suppose that $\Pi^{(d,s)}(k)$ and $\Xi^{(d,s)}(k)$ are respectively a $2^n \times 1$ vector and $2^n \times 2^n$ matrix whose entries are given in (36) and (37). Then,

$$\Pi^{(d,s)}(k) = \frac{\Gamma^{(d,s)}(k)\circ\Delta^{(d,s)}(k)}{\parallel \Gamma^{(d,s)}(k)\circ\Delta^{(d,s)}(k)\parallel_1},\ k=1,\ldots,m, \tag{42}$$

$$\Xi^{(d,s)}(k) = \frac{\Big[\Gamma^{(d,s)}(k)\Delta^{(d,s)}(k+1)^T\Big]\circ A^{(s)}\circ\mathbf{\Phi}^{(d,s)}(k+1)}{\parallel \Gamma^{(d,s)}(m)\parallel_1}, \tag{43}$$

$k=1,\ldots,m-1$, where $\mathbf{\Phi}^{(d,s)}(k)$ is the $2^n\times 2^n$ matrix

$$\mathbf{\Phi}^{(d,s)}(k) = \Big[\Phi^{(d,s)}(k),\ldots,\Phi^{(d,s)}(k)\Big]^T. \tag{44}$$

In the case of missing data, if there is a time point k at which no expression of the n genes is observed ($k\notin T_{obs}$), then $p(\mathbf{Y}_k|\mathbf{X}_k=\mathbf{x}^i)=1$ for any $i=1,\ldots,2^n$, and $\Phi(k)=\mathbf{1}_{2^n}$ in Fig. 1, as well as in all the Equations (36), (37),(38), (39), (40), (41), (42), (43) and (44) ; however, if at least one gene expression is observed at time k ($k\in T_{obs}$), then $p(\mathbf{Y}_k|\mathbf{X}_k)$, and thus $\Phi_j(k)=p(\mathbf{Y}_k|\mathbf{X}_k=\mathbf{x}^j)$, is computed from (10).

3.2 Learning Algorithm

In the previous section, we demonstrated that, given the network function, we can estimate the parameters by the EM method. Let $\hat{\theta}_i$ denote the estimated parameter vector, defined in (5) and derived via the EM algorithm when $\mathbf{f}=\mathbf{f}^i$ for $i=1,\ldots,M$. The final estimates for both the network function, $\hat{\mathbf{f}}$, and the parameter vector, $\hat{\theta}$, can be determined from (6). Let $l(\mathbb{Y}|\mathbf{f},\theta)=\log p(\mathbb{Y}|\mathbf{f},\theta)$ be the log-likelihood of the observed trajectories. Since all D observations are independent,

$$l(\mathbb{Y}|\mathbf{f},\theta)=\sum_{d=1}^{D}\log p(\mathcal{Y}^{(d)}|\mathbf{f},\theta). \tag{45}$$

For $d=1,\ldots,D$, $p(\mathcal{Y}^{(d)}|\mathbf{f},\theta)$ is derived by marginalizing the joint distribution of the states and observations over the states as

$$p(\mathcal{Y}^{(d)}|\mathbf{f},\theta)=\sum_{\mathcal{X}^{(d)}}p(\mathcal{X}^{(d)},\mathcal{Y}^{(d)}|\mathbf{f},\theta), \tag{46}$$

where $p(\mathcal{X}^{(d)},\mathcal{Y}^{(d)}|\mathbf{f},\theta)$ is given by (8) with the factor-graph shown in Fig. 1. Computing $p(\mathcal{Y}^{(d)}|\mathbf{f},\theta)$ only requires the forward parameter Γ and recursions up to time m. In fact, the desired likelihood for the dth trajectory is the summation of the entries of $\Gamma^{(d)}(m)$, that is,

$$p(\mathcal{Y}^{(d)}|\mathbf{f},\theta)=\parallel \Gamma^{(d)}(m)\parallel_1, \tag{47}$$

where $\Gamma^{(d)}(m)$ can be computed in the same way as in (40), assuming the parameter vector θ and the network function $\mathbf{f}$. From (45) and (47), we can write

$$l(\mathbb{Y}|\mathbf{f},\theta)=\sum_{d=1}^{D}\log\ \parallel \Gamma^{(d)}(m)\parallel_1 . \tag{48}$$

Then, according to (6), the final estimate for the network function and parameters is given by

$$(\hat{\mathbf{f}},\hat{\theta})=\arg\max_{(\mathbf{f},\theta)\in\{(\mathbf{f}^1,\hat{\theta}_1),\ldots,(\mathbf{f}^M,\hat{\theta}_M)\}} l(\mathbb{Y}|\mathbf{f},\theta). \tag{49}$$

We summarize the algorithm for learning the network function and model parameters in Algorithm 1.

Algorithm 1. Learning Algorithm

Inputs: the number of genes n, the length of trajectories m, the set of D observations $\mathbb{Y}=\{\mathcal{Y}^{(1)},\mathcal{Y}^{(2)},\cdots,\mathcal{Y}^{(D)}\}$, the set of uncertain network functions $\mathbf{F}=\{\mathbf{f}^1,\ldots,\mathbf{f}^M\}$, and a convergence threshold, τ, for the EM.
Outputs: $\hat{\mathbf{f}}$ and $\hat{\theta}$, the estimated network function and model parameters $\theta=[p,\lambda,\delta,\sigma^2]^T$.
procedure
  for $i=1$ to M **do**
    • $s=0$
    • $\mathbf{f}=\mathbf{f}^i$
    • Initialize $\theta^{(0)}=0$
    • Randomly initialize $\theta^{(1)}$
    while $\parallel \theta^{(s+1)}-\theta^{(s)}\parallel > \tau$ **do**
      • $s=s+1$
      E-step:
      • Compute $\Pi^{(d,s)}(k)$ for any $k=1,\ldots,m$, and $d=1,\ldots,D$ via (42).
      • Compute $\Xi^{(d,s)}(k)$ for any $k=1,\ldots,m-1$, and $d=1,\ldots,D$ via (43).
      M-step:
      • Compute ρ_j for $j=1,\ldots,6$, via (25), (26), (27), (28), (32), and (35).
      • Compute $\lambda^{(s+1)}$ via (30).
      • Compute $\delta^{(s+1)}$ via (31).
      • Compute $\sigma^{2(s+1)}$ via (33).
      • Compute $p^{(s+1)}$ via (34).
      • $\theta^{(s+1)}=\Big[p^{(s+1)},\lambda^{(s+1)},\delta^{(s+1)},\sigma^{2(s+1)}\Big]^T$
    end while
    • $\hat{\theta}_i=\theta^{(s+1)}$
  end for
  • Get $\hat{\mathbf{f}}$ and $\hat{\theta}$ via (48) and (49).
end procedure

3.3 Plug-In Bayes Classifier

Let labels 0 and 1 refer to the healthy and mutated classes, respectively, let $\mathbb{Y}_0$ and $\mathbb{Y}_1$ denote the respective training trajectory sets, and let $\mathbf{F}_0$ and $\mathbf{F}_1$ denote the respective uncertain network function sets for the two classes. We apply Algorithm 1 to both classes, with corresponding $\mathbb{Y}$ and $\mathbf{F}$, to derive the learned network functions $\hat{\mathbf{f}}_0$ and $\hat{\mathbf{f}}_1$, and the estimated parameters $\hat{\theta}_0$ and $\hat{\theta}_1$, for the healthy and mutated classes, respectively. These are plugged into the Bayes classifier. For any new trajectory $\mathcal{Y}=[\mathbf{Y}_1,\ldots,\mathbf{Y}_m]$ of n genes with any arbitrary length m and possibly missing data, the

classifier is defined by

$$\psi_D(\mathcal{Y}) = \begin{cases} 1, & \hat{c}_1 p(\mathcal{Y}|\hat{\mathbf{f}}_1, \hat{\theta}_1) \geq \hat{c}_0 p(\mathcal{Y}|\hat{\mathbf{f}}_0, \hat{\theta}_0) \\ 0, & \hat{c}_1 p(\mathcal{Y}|\hat{\mathbf{f}}_1, \hat{\theta}_1) < \hat{c}_0 p(\mathcal{Y}|\hat{\mathbf{f}}_0, \hat{\theta}_0), \end{cases} \tag{50}$$

where $\hat{c}_0$ and $\hat{c}_1$ are the estimated values of the prior probabilities of the healthy and mutated classes, respectively. These can be estimated by the number of training trajectories for each class divided by the total number of training trajectories D; however, this estimate is unreliable for small samples [29]. Often there are substantial data regarding the proportions of healthy and pathological phenotypes–for instance, false negative rates resulting from preliminary testing such as mammography and needle biopsies, so that excellent estimates of $\hat{c}_0$ and $\hat{c}_1$ are available for genomic classification following preliminary testing. We assume the equiprobable case, $\hat{c}_0 = \hat{c}_1 = \frac{1}{2}$, which makes classification most challenging.

In the simulations we generate an equal number of training trajectories for each class, $|\mathbb{Y}_0| = |\mathbb{Y}_1| = \frac{D}{2}$. The likelihoods $p(\mathcal{Y}|\hat{\mathbf{f}}_0, \hat{\theta}_0)$ and $p(\mathcal{Y}|\hat{\mathbf{f}}_1, \hat{\theta}_1)$ in (50) can be computed, as in (47), by

$$p(\mathcal{Y}|\hat{\mathbf{f}}_i, \hat{\theta}_i) = \parallel \Gamma^{(i)}(m) \parallel_1, \ i = 0, 1, \tag{51}$$

where $\Gamma^{(i)}(m)$ can be computed by forward computations, as in (40), by

$$\begin{aligned} \Gamma^{(i)}(1) &= {\pi^{(i)}}^T \circ \Phi^{(i)}(1), \\ \Gamma^{(i)}(k+1) &= \left[{A^{(i)}}^T \Gamma^{(i)}(k) \right] \circ \Phi^{(i)}(k+1), \end{aligned} \tag{52}$$

where $\pi^{(i)}$, $A^{(i)}$, and $\Phi^{(i)}$ are computed for the class $i = 0, 1$, assuming $\mathbf{f} = \hat{\mathbf{f}}_i$ and $\theta = \hat{\theta}_i$.

4 Classification of Averaged Steady-State Expression Data in Multiple-Cell Scenarios

In the absence of single-cell technology, when measuring expressions in a nonsynchronized multiple-cell setting, at each time point the measured expression value of each gene is an average over 2^n different states. The underlying state model for the evolution of the genes in each cell is the same BNp model (1); however, since the observations are static expression data in the steady-state and not trajectories, we use the state model (1) only for calculating the steady-state distribution π.

4.1 Observation Model

Suppose that the expression values of the genes in every individual are measured from a tissue consisting of N cells. As mentioned previously, the variability existing in the different cells of an individual is inter-cell variability. According to (4), in every cell c of an individual, the expression values of the n genes, $\mathbf{Y}^{(c)} = [y_1^{(c)}, \ldots, y_n^{(c)}]^T$, given the state $\mathbf{X} = [x_1, \ldots, x_n]^T$ in the steady-state, follow a Gaussian model

$$p(\mathbf{Y}^{(c)}|\mathbf{X}) \sim \mathcal{N}\big(\lambda \mathbf{1}_n + \delta \mathbf{X}, \sigma_{ic}^2 I_n\big), \ c = 1, \ldots, N, \tag{53}$$

where σ_{ic}^2 denotes the inter-cell variability across the different cells of an individual. In the multiple-cell scenario we do not observe expression values of the genes in every single cell but only observe the expression averaged over N cells, namely,

$$S_N = \frac{\sum_{c=1}^N \mathbf{Y}^{(c)}}{N}. \tag{54}$$

We can obtain the distribution of $\mathbf{Y}^{(c)}$ from (53) by marginalizing over the states $\mathbf{X}$, which have the steady-state distribution π. Doing so, the distribution of $\mathbf{Y}^{(c)}$ for any c is

$$p(\mathbf{Y}^{(c)}) = \sum_{i=1}^{2^n} p(\mathbf{Y}^{(c)}|\mathbf{X} = \mathbf{x}^i)\pi_i, \tag{55}$$

which is a Gaussian mixture distribution with 2^n components. Since the $\mathbf{Y}^{(c)}$'s are independent for different cells and have the same mean and covariance matrix, we can use the central-limit theorem to approximate the distribution of S_N. The mean and covariance matrix of $\mathbf{Y}^{(c)}$ are given by

$$\begin{aligned} \mu &= \mathrm{E}\Big[\mathbf{Y}^{(c)}\Big] = \mathrm{E}\Big[\mathrm{E}\Big[\mathbf{Y}^{(c)}|\mathbf{X}\Big]\Big] \\ &= \mathrm{E}[\lambda 1_n + \delta \mathbf{X}] = \lambda \mathbf{1}_n + \delta \sum_{i=1}^{2^n} \mathbf{x}^i \pi_i, \end{aligned} \tag{56}$$

$$\begin{aligned} \Sigma_{\mathbf{Y}} &= \mathrm{cov}\Big(\mathbf{Y}^{(c)}\Big) = \mathrm{cov}\Big(\mathrm{E}(\mathbf{Y}^{(c)}|\mathbf{X})\Big) + \mathrm{E}(\mathrm{cov}\Big(\mathbf{Y}^{(c)}|\mathbf{X}\Big) \\ &= \mathrm{cov}(\lambda \mathbf{1}_n + \delta \mathbf{X}) + \mathrm{E}(\sigma_{ic}^2 I_n) = \delta^2 \Sigma_{\mathbf{X}} + \sigma_{ic}^2 I_n, \end{aligned} \tag{57}$$

where $\Sigma_{\mathbf{X}}$ is the covariance matrix of the states in the steady-state,

$$\begin{aligned} \Sigma_{\mathbf{X}} &= \mathrm{cov}(\mathbf{X}) = \mathrm{E}(\mathbf{X}\mathbf{X}^T) - \mathrm{E}(\mathbf{X})\mathrm{E}(\mathbf{X})^T \\ &= \sum_{i=1}^{2^n} \mathbf{x}^i {\mathbf{x}^i}^T \pi_i - \left(\sum_{i=1}^{2^n} \mathbf{x}^i \pi_i \right) \left(\sum_{i=1}^{2^n} \mathbf{x}^i \pi_i \right)^T. \end{aligned} \tag{58}$$

According to the central-limit theorem, when N is large (having many cells), the distribution of S_N converges to the multivariate normal

$$S_N \sim \mathcal{N}\left(\mu, \frac{\Sigma_{\mathbf{Y}}}{N}\right), \tag{59}$$

where μ and $\Sigma_{\mathbf{Y}}$ are given in (56) and (57).

Since S_N is the averaged expression values of the genes over many cells in only one individual and our samples come from different individuals, we should also take into account the inter-subject variability between different individuals. As a result, in the multiple-cell scenario, the expression values of the n genes, denoted by the vector $\mathbf{Z} = [z_1, \ldots, z_n]^T$, can be modeled by

$$\mathbf{Z} = S_N + \epsilon, \tag{60}$$

where ϵ provides the inter-subject variability. Since in the single-cell scenario we assumed that $\epsilon \sim N(0, \sigma^2 I_n)$, we assume the same here. Since S_N and ϵ are multivariate Gaussian and independent of each other, $\mathbf{Z}$ has a multivariate Gaussian distribution,

$$p(\mathbf{Z}) \sim \mathcal{N}\left(\mu, \frac{\Sigma_{\Upsilon}}{N} + \sigma^2 I_n\right). \tag{61}$$

Usually there are many cells (large N) in the tissues from which the expressions are measured. N is typically large in practice, for instance, according to [30], there are millions of cells in bulk RNA-Seq experiments. Hence, the first part of the covariance matrix of $\mathbf{Z}$, that is, $\frac{\Sigma_{\Upsilon}}{N}$, has negligible entries, and we can well approximate the distribution of $\mathbf{Z}$ by

$$p(\mathbf{Z}|\mu, \sigma^2) \sim \mathcal{N}\left(\mu = \lambda \mathbf{1}_n + \delta \sum_{i=1}^{2^n} \mathbf{x}^i \pi_i, \sigma^2 I_n\right). \tag{62}$$

4.2 Plug-In Bayes Classifier

To use the Bayes plug-in classifier, we need to estimate the parameters using the training data, but since the class-conditional densities in (62) are Gaussian and σ^2 may be different in the two classes, the Bayes plug-in classifier is quadratic discriminant analysis (QDA). Hence, we need only estimate μ and σ^2 in (62), not λ, δ, and π. This reduces the complexity because we skip the cumbersome optimization problem of finding p (for estimating π), which would have been done for each of M possible network functions $\mathbf{f}^{(i)}$, for $i = 1, \ldots, M$, for each class. Moreover, we do not even need to partially know the network functions in the classifier, which is beneficial when we have no knowledge of network structures. In other words, although μ in (62) is a function of λ, δ , p, and $\mathbf{f}$ (two last determine π), we do not need to estimate them to estimate μ, which we can directly estimate from the observed data.

The ML estimates of μ and σ^2 in (62) from observed data $\mathbb{Z} = [\mathbf{Z}^{(1)}, \ldots, \mathbf{Z}^{(D)}]$ are

$$\hat{\mu} = \frac{\sum_{d=1}^{D} \mathbf{Z}^{(d)}}{D}, \tag{63}$$

$$\hat{\sigma}^2 = \frac{\sum_{d=1}^{D} \parallel \mathbf{Z}^{(d)} - \hat{\mu} \parallel_2^2}{nD}, \tag{64}$$

respectively. The log-likelihood (after dropping the constant parts) of any expression vector $\mathbf{Z}$ from (62) is

$$l(\mathbf{Z}|\mu, \sigma^2) = -\frac{n}{2} \log \sigma^2 - \frac{\parallel \mathbf{Z} - \mu \parallel_2^2}{2\sigma^2}. \tag{65}$$

Let $\mathbb{Z}_0$ and $\mathbb{Z}_1$ denote the training data sets of the healthy and mutated classes, respectively, the total number of data points being D. Suppose $\hat{\mu}_i$ and $\hat{\sigma}_i^2$ are the estimated values (using (63) and (64)) for the class $i = 0, 1$ using $\mathbb{Z}_i$. We assume $\hat{c}_0 = \hat{c}_1 = \frac{1}{2}$ and $|\mathbb{Z}_0| = |\mathbb{Z}_1| = \frac{D}{2}$. The Bayes plug-in classifier (QDA) for $\mathbf{Z} = [z_1, \ldots, z_n]^T$ is

$$\psi_D(\mathbf{Z}) = \begin{cases} 1, & l(\mathbf{Z}|\hat{\mu}_1, \hat{\sigma}_1^2) \geq l(\mathbf{Z}|\hat{\mu}_0, \hat{\sigma}_0^2) \\ 0, & l(\mathbf{Z}|\hat{\mu}_1, \hat{\sigma}_1^2) < l(\mathbf{Z}|\hat{\mu}_0, \hat{\sigma}_0^2), \end{cases} \tag{66}$$

where $l(\mathbf{Z}|\hat{\mu}_i, \hat{\sigma}_i^2)$ can be computed from (65) for each class $i = 0, 1$.

4.3 Classification Difficulty

We wish to quantify classification difficulty relative to attractor structure. From (62), the expression values of the n genes in classes 0 and 1 are modeled as

$$p(\mathbf{Z}^{(j)}) \sim \mathcal{N}\left(\mu_j, \sigma_j^2 I_n\right), \ j = 0, 1. \tag{67}$$

If we assume that λ and δ are the same in the two classes, then

$$\mu_j = \lambda 1_n + \delta \sum_{i=1}^{2^n} \mathbf{x}^i \pi_i^{(j)} = \lambda 1_n + \delta X \pi^{(j)T}, \ j = 0, 1, \tag{68}$$

where $X = [\mathbf{x}^1, \ldots, \mathbf{x}^{2^n}]$ is the $n \times 2^n$ binary matrix representing the binary states, its i th column being the i th binary state.

In the Gaussian settings, the means and covariance matrices affect classification error. We focus on the distance

$$\begin{aligned} \xi &= (\mu_0 - \mu_1)^T (\mu_0 - \mu_1) \\ &= \delta^2 \left(\pi^{(0)} - \pi^{(1)}\right) X^T X \left(\pi^{(0)} - \pi^{(1)}\right)^T \end{aligned} \tag{69}$$

between the means because this distance is directly relatable to the attractors. If the perturbation probability p is very small and each class has only one attractor cycle, then the steady-state distributions are accurately approximated by

$$\pi_i^{(j)} = \frac{1}{|\mathcal{A}_j|} 1(i \in \mathcal{A}_j), \ j = 0, 1, \ i = 1, \ldots, 2^n, \tag{70}$$

where $\mathcal{A}_j$ is the set of the attractor states of class j, $1(.)$ is the indicator function (equals to 1 if its argument is true and equals to 0 otherwise), and $|\mathcal{A}_j| \in \{1, \ldots, 2^n\}$ is the attractor length for class j [31].

For any attractor lengths we find networks having minimum ($\xi = 0$) and maximum distances, where a network is identified with its attractor states because according to the preceding equation ξ depends only on these. Letting $\Sigma(\mathcal{A}_j)$ denote the sum of the values in $\mathcal{A}_j$, if $|\mathcal{A}_0| = |\mathcal{A}_1|$ and $\Sigma(\mathcal{A}_0) = \Sigma(\mathcal{A}_1)$, then $\xi = 0$. We represent minimum ξ and maximum ξ cases by $(\mathcal{A}_0^f, \mathcal{A}_1^f)$ (f for failure) and $(\mathcal{A}_0^o, \mathcal{A}_1^o)$ (o for optimal), respectively, where

$$(\mathcal{A}_0^f, \mathcal{A}_1^f) = \left\{\mathcal{A}_0, \mathcal{A}_1 : \left(\pi^{(0)} - \pi^{(1)}\right) X^T X \left(\pi^{(0)} - \pi^{(1)}\right)^T = 0\right\}, \tag{71}$$

$$(\mathcal{A}_0^o, \mathcal{A}_1^o) = \arg \max_{\mathcal{A}_0, \mathcal{A}_1} \left(\pi^{(0)} - \pi^{(1)}\right) X^T X \left(\pi^{(0)} - \pi^{(1)}\right)^T. \tag{72}$$

Letting $l_0 = |\mathcal{A}_0|$ and $l_1 = |\mathcal{A}_1|$ denote the lengths, the numbers of sets $\mathcal{A}_0$ and $\mathcal{A}_1$ are $\binom{2^n}{l_0}$ and $\binom{2^n}{l_1}$, respectively. Note that there are $(l_0 - 1)!$ and $(l_1 - 1)!$ cyclic permutations of each set, which lead to different attarctors, but we do not consider them since they all have the same average and thus are equivalent in the current sense. Hence, the size of the search space for solving (71) and (72) is $\binom{2^n}{l_0} \binom{2^n}{l_1}$. In general, the solutions are not unique.

5 Simulation Results and Discussion

As this is the first paper which studies supervised classification of single-cell gene expression trajectories under the framework of Boolean networks with perturbations, there is not a similar trajectory-based method to compare it with. However, we compare the performance of our proposed trajectory-based classifier, using single-cell gene expression

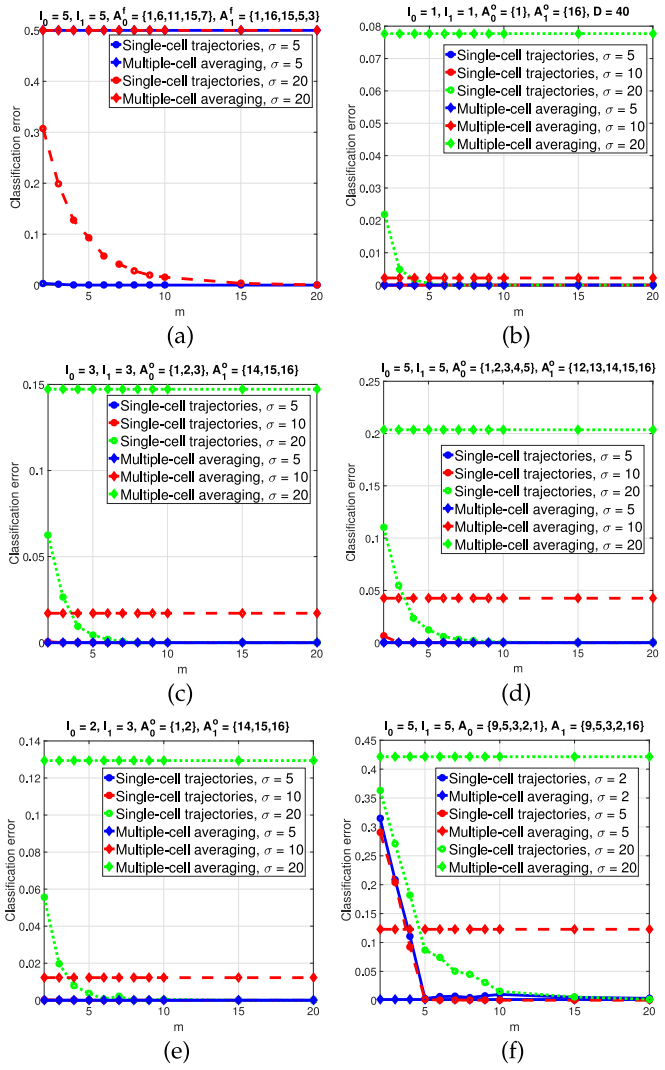

Fig. 2. Classification error of single-cell trajectory method versus m. The classification error of the multiple-cell averaging method is also included in the plots for comparison.

trajectories as its input, with that of a multiple-cell averaging classifier which uses bulk gene expression data, like RNA-Seq or microarray, as its input. We have set the model parameters to do a fair comparison between these two methods. The comparisons show a clear advantage of the first method, especially in high-noise scenarios.

5.1 Some Specific Networks

Let $\mathcal{A} = \{a_1, a_2, \ldots, a_l\}$ be the set of attractor states with the length l, in which order matters, such that the attractor cycle is $a_1 \to a_2 \to \cdots \to a_l \to a_1$. We consider three specific cases as examples and compare the two methods of classification, single-cell trajectories and multiple-cell averaging. In all the cases, we assume $n = 4$, $p = 0.001$, $p_{miss} = 0$, $\lambda = 10$, and $\delta = 30$.

Case 1. Suppose the failure case with $l_0 = 5$ and $l_1 = 5$. We choose $\mathcal{A}_0 = \{1, 6, 11, 15, 7\}$ and $\mathcal{A}_1 = \{1, 16, 15, 5, 3\}$. Since $|\mathcal{A}_0| = |\mathcal{A}_1|$ and $\Sigma(\mathcal{A}_0) = \Sigma(\mathcal{A}_1)$, $\xi = 0$. Hence, we expect that averaging cannot perform well. Fig. 2a represents the classification error of the two methods, single-cell trajectories and multiple-cell averaging, for $\sigma = 5$ and $\sigma = 20$. It can be seen that for any value of σ, averaging has the maximum classification error 0.5. However, the single-cell trajectory method has the error 0 (perfect classification for all m) for $\sigma = 5$, and a decreasing error as a function of m for $\sigma = 20$.

Case 2. Figs. 2b, 2c, 2d, and Fig. 2e, which show results for optimal attractor sets derived from (72), relate to $l_0 = l_1 = 1$ (single attractors), $l_0 = l_1 = 3$, $l_0 = l_1 = 5$, and $l_0 = 2, l_1 = 3$, respectively. There are many solutions to (72). We only select the first solution to show the results, the selected optimal attractor sets $\mathcal{A}_0$ and $\mathcal{A}_1$ being written at the top of each figure. For example, in Fig. 2b ($l_0 = l_1 = 1$), $\mathcal{A}_0 = \{1\}$ and $\mathcal{A}_1 = \{16\}$, meaning that the attractor cycles in class 0 and class 1 are $[0,0,0,0]^T \to [0,0,0,0]^T$ and $[1,1,1,1]^T \to [1,1,1,1]^T$, respectively. For the low noise levels ($\sigma = 5$), both methods yield zero classification error. For $\sigma = 10$, the trajectory method still has zero error for every m, but averaging has nonzero error, even though its difference with zero is slight. For $\sigma = 20$, the trajectory method has nonzero error for small m, but it is still much less than the error of the averaging method. No matter the size of σ, the error of the trajectory method will converge to 0 for sufficiently large m. In sum, the trajectory method is more robust realtive to the noise level than the averaging method.

Case 3. The only scenario in which averaging can work better than the trajectory method is when there are similar trajectories in the attractor cycles of the two classes. In such situations, the trajectory method may make a mistake in classifying short trajectories, while the averaging method may be able to classify better. For example, consider $l_0 = l_1 = 5$ with $\mathcal{A}_0 = \{9, 5, 3, 2, 1\}$ and $\mathcal{A}_1 = \{9, 5, 3, 2, 16\}$. The trajectory $9 \to 5 \to 3 \to 2$ exists in the both attractor cycles. As a result, we expect that for short observed trajectories, the trajectory method will perform poorly. Fig. 2f shows the results in this case for $\sigma = 2, 5, 20$. For $\sigma = 2$, the averaging method yields zero error but the trajectory method has nonzero error for $m \leq 4$. For $\sigma = 5$, the averaging method still has better performance than the trajectory method with $m \leq 3$, but for $\sigma = 20$, the trajectory method outperforms averaging for any m. This again shows that the averaging method cannot work in high noise regimes, whereas the trajectory method can result in a very low error if the observed trajectories are long enough.

5.2 Random Synthetic Networks

To evaluate performance on random synthetic networks, we randomly generate 500 Boolean networks for each case of $n = 4, 6, 8$ genes and consider a maximum in-degree of $K = 2$, meaning that each gene has 1 or 2 randomly assigned predictors. If the ith gene has in-degree K_i, its 2^{K_i} outputs are selected from a Bernoulli distribution with parameter $p_{bias} = 0.5$. A single-bit mutation is applied to all healthy networks to obtain the corresponding 500 mutated networks. Specifically, we randomly pick a gene, say gene i, and randomly flip the value of one of its 2^{K_i} outputs, $0 \to 1$ and $1 \to 0$. This mutation changes the output of 2^{n-K_i} states in the truth table of the healthy BN. We restrict the generated healthy and mutated BNs to have a single attractor cycle, with the minimum length of the two attractor cycles being L. The simulations will demonstrate that L is an important parameter in determining the sufficient trajectory length in low-noise scenarios. In all simulations, we use the

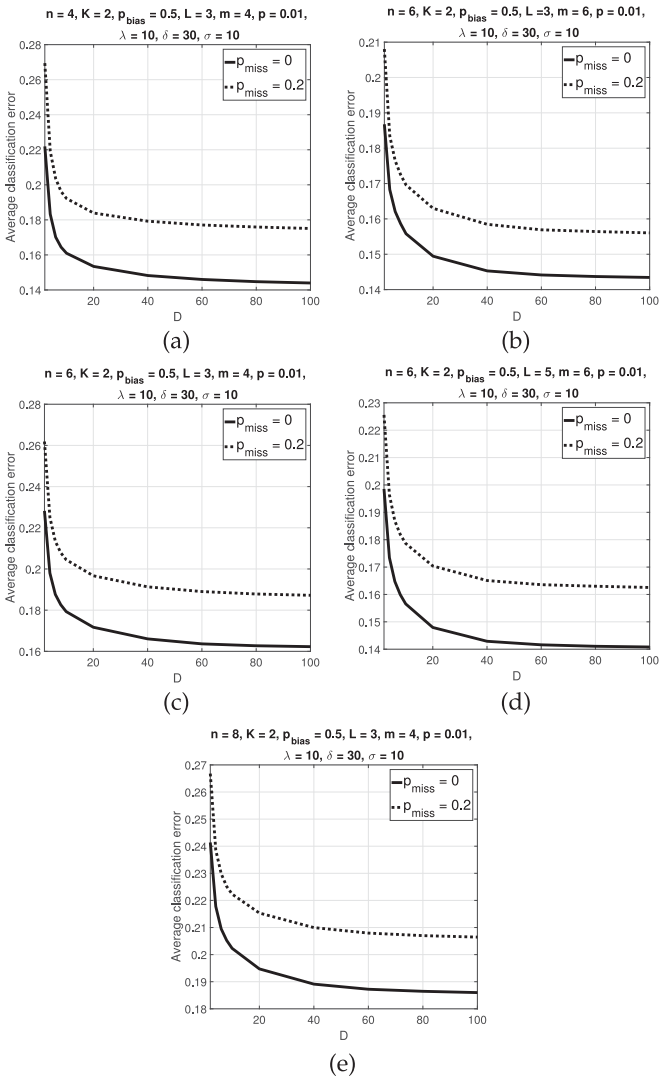

Fig. 3. Average classification error of the trajectory classifier over 500 synthetic BNs versus D with $K = 2$ and $p_{bias} = 0.5$. Parameter values are $p = 0.01$, $\lambda = 10$, $\delta = 30$, $\sigma = 10$, (a) $n = 4$, $L = 3$, $m = 4$, (b) $n = 4$, $L = 5$, $m = 6$, (c) $n = 6$, $L = 3$, $m = 4$, (d) $n = 6$, $L = 5$, $m = 6$, and (e) $n = 8$, $L = 3$, $m = 4$.

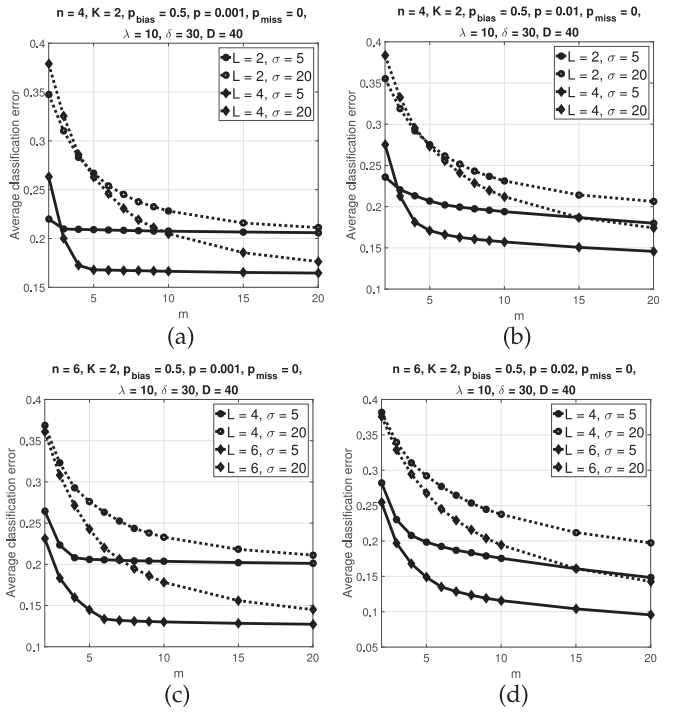

Fig. 4. Average classification error of the trajectory classifier over 500 synthetic BNs versus m with $K = 2$ and $p_{bias} = 0.5$. Parameter values are $p_{miss} = 0$, $\lambda = 10$, $\delta = 30$, $D = 40$, (a) $n = 4$, $p = 0.001$, (b) $n = 4$, $p = 0.01$, (c) $n = 6$, $p = 0.001$, and (d) $n = 6$, $p = 0.02$.

same parameter values, p, λ, δ, σ^2, for both the healthy and mutated networks: $\lambda = 10$, $\delta = 30$, and $M = 2$. We use three different values for the observation noise level: $\sigma = 5$ (low noise), $\sigma = 10$ (medium noise), and $\sigma = 20$ (high noise).

Fig. 3, in which $\sigma = 10$, shows average classification error versus D, the total number of training trajectories for both the healthy and mutated BNs, for different numbers of genes n, trajectory length m, minimum attractor length L, gene perturbation probability p, and gene missing probability p_{miss}. As expected, missing observations deteriorate classifier performance in all cases. Average error decreases with more training trajectories and converges to a fixed value when D becomes large enough. The value of D required for a converged error rate depends on n, m, and p_{miss}. For a given network size, having larger m and lower p_{miss} can speed up estimation of the network parameters, so that smaller D is required to achieve the converged error. Furthermore, for larger networks, the required value of D increases. Note that we have assumed that we partially know the networks, so that the search space is limited to M functions. Hence, the error curves have fairly fast convergence realtive to D. In the absence of network knowledge, more training data would be required to correctly learn the networks and convergence would be slower.

Fig. 4 demonstrates the behavior of the average classification error versus m, where based on the results in Fig. 3, we assume $D = 40$ training trajectories to achieve the converged error. We set $p_{miss} = 0$ in Fig. 4. Figs. 4a and 4b show the error for $n = 4$ gene networks when $p = 0.001$ and $p = 0.01$, respectively, assuming different values of $L = 2, 4$ and $\sigma = 5, 20$. Figs. 4c and 4d present the error for $n = 6$ gene networks when $p = 0.001$ and $p = 0.02$, respectively, assuming different values of $L = 4, 6$ and $\sigma = 5, 20$. In all figures for every value of m and L, the error increases with increasing σ. Moreover, the error curves are always monotonically decreasing in terms of m. There is a special case in which the error gets fixed after some m. This is when p is close to 0 and σ is small. In such conditions, the sufficient m to achieve the least possible error is $L + 1$ because when $p \approx 0$ the BNps tend to BNs, which are deterministic, meaning that the observations occur only in the attractor states and circulate inside the attractor cycles. In such a case, the maximum length of a trajectory that can help distinguish the two networks is $L + 1$, where L is the minimum length of the attractor cycles in the two networks. When p is considerable, there is a nonnegligible probability of jumping states, so that longer trajectories can help. In Figs. 4a and 4c, where $p = 0.001$, when $\sigma = 5$, the error curves flatten out after $m = L + 1 = 3, 5, 7$, corresponding to $L = 2, 4, 6$, respectively. In Figs. 4a and 4c, in which observation noise is high, $\sigma = 20$, the error curves still converge to a constant value, but the convergence is much slower and longer trajectories are required ($> L + 1$). In Figs. 4b and 4d, where

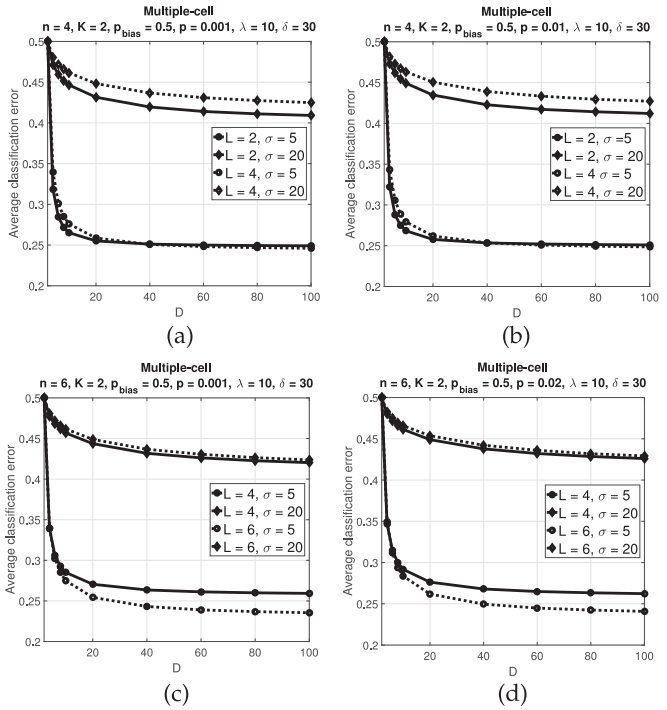

Fig. 5. Average classification error of the multiple-cell classifier over 500 synthetic BNs versus D with $K = 2$ and $p_{bias} = 0.5$. Parameter values are $\lambda = 10$, $\delta = 30$, (a) $n = 4$, $p = 0.001$, (b) $n = 4$, $p = 0.01$, (c) $n = 6$, $p = 0.001$, and (d) $n = 6$, $p = 0.02$.

$p = 0.01, 0.02,$ respectively, the error curves are permanently decreasing with increasing m and do not converge to a fixed value, even in the low-noise cases.

Fig. 5 depicts the average errors versus D in multiple-cell scenarios, where there is no trajectory data but only averaged expression data. Figs. 5a and 5b show the error curves in 4-gene networks for $p = 0.001$ and $p = 0.01$, respectively, and different values of $L = 2, 4$ and $\sigma = 5, 20$. Figs. 5c and 5d show similar results in 6-gene networks for $p = 0.001$ and $p = 0.02$, respectively, and different values of $L = 4, 6$ and $\sigma = 5, 20$. The error is higher in high-noise cases and it decreases to converge to a fixed value. The convergence rate depends on σ. When σ is low (high), the convergence is fast (slow).

Fig. 6 shows average classification error versus m for different values of $p_{miss} = 0$ (no missing), $p_{miss} = 0.2$ (low missing probability), and $p_{miss} = 0.5$ (high missing probability). For the sake of comparison, we have also included in Fig. 6 the error of the multiple-cell scenarios for the same parameter values. Fig. 6a shows the results for 4-gene networks when $L = 4$, $p = 0.001$, $\sigma = 5$, and $D = 40$. In this figure, smaller p_{miss} always yields a lower error rate for every value of m, but the differences decrease as m grows. The salient point of Fig. 6a is that one can always get a lower error rate by using the single-cell trajectory data, even with missing data, than by using the multiple-cell averaged data. In the case of Fig. 6a, the trajectory data with $p_{miss} = 0, 0.2, 0.5$ has lower error than multiple-cell data when $m \geq 3$, $m \geq 3$, $m \geq 5$, respectively. We previously mentioned that when $p \approx 0$ and σ is small, the error curves flatten out after $m = L + 1$; Fig. 6a shows that that is true when there are no missing data ($p_{miss} = 0$). With missing data, convergence is slow and longer trajectories are required to reach the converged error. Fig. 6b shows similar results for 6-gene networks when $L = 6$, $p = 0.01$, $\sigma = 10$, and $D = 40$. Again, classification using trajectory data, even with high probability of missing data, can considerably lower the error rate as opposed to using multiple-cell averaged data.

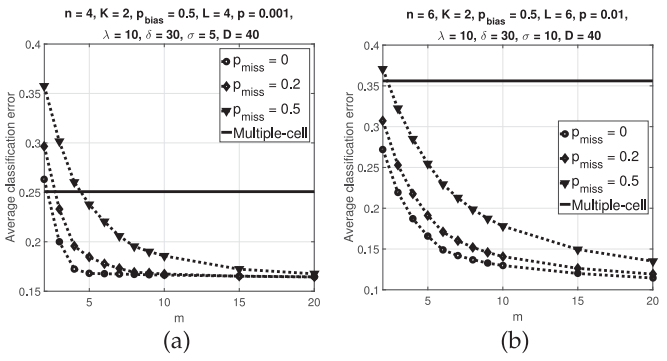

Fig. 6. Average classification error of the trajectory classifier and multiple-cell classifier over 500 synthetic BNs versus m with $K = 2$ and $p_{bias} = 0.5$. Parameter values are $\lambda = 10$, $\delta = 30$, $D = 40$, (a) $n = 4$, $L = 4$, $p = 0.001$, $\sigma = 5$, and (b) $n = 6$, $L = 6$, $p = 0.01$, $\sigma = 10$.

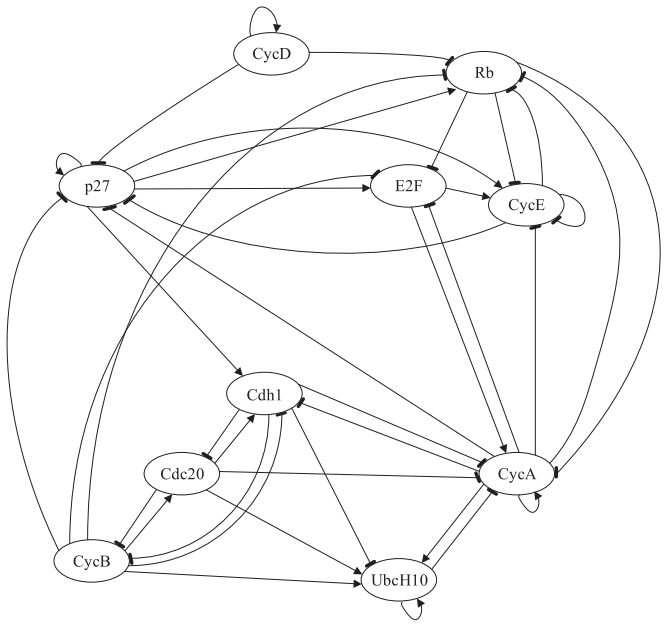

Fig. 7. Mammalian cell-cycle gene regulatory network.

5.3 Real Network: Mammalian Cell-Cycle BN

For an illustration using a real network, we use the wild-type mammalian cell-cycle BN, whose GRN [32] is shown in Fig. 7. This GRN has ten genes. The regulating functions are defined in Table 1 [32]. According to [32], one mutated situation is that the gene p27 is always off and cannot be activated. As a result, we derive the healthy BN from Table 1 and for the mutated/cancerous BN we put the value of p27 in Table 1 to zero, that is, $f_3 = 0$. This means that in the cancerous scenario the value of p27 does not obey the regulating functions and is always zero. The gene CycD is determined by extracellular signals. As we do not know the value of CycD, we have $M = 2$ candidate network functions for each of the healthy and mutated networks, which are

TABLE 1
Definitions of Boolean Functions for the Wild-Type Mammalian Cell-Cycle BN with 10 Genes

Order	Gene	Regulating function
x_1	$CycD$	$f_1 =$ Extracellular signals
x_2	Rb	$f_2 = (\overline{CycD} \wedge \overline{CycE} \wedge \overline{CycA} \wedge \overline{CycB}) \vee (p27 \wedge \overline{CycD} \wedge \overline{CycB})$
x_3	$p27$	$f_3 = (\overline{CycD} \wedge \overline{CycE} \wedge \overline{CycA} \wedge \overline{CycB}) \vee (p27 \wedge \overline{(CycE \wedge CycA)} \wedge \overline{CycD} \wedge \overline{CycB})$
x_4	$E2F$	$f_4 = (\overline{Rb} \wedge \overline{CycA} \wedge \overline{CycB}) \vee (p27 \wedge \overline{Rb} \wedge \overline{CycB})$
x_5	$CycE$	$f_5 = (E2F \wedge \overline{Rb})$
x_6	$CycA$	$f_6 = (E2F \wedge \overline{Rb} \wedge \overline{Cdc20} \wedge \overline{(Cdh1 \wedge UbcH10)}) \vee (CycA \wedge \overline{Rb} \wedge \overline{Cdc20} \wedge \overline{(Cdh1 \wedge UbcH10)})$
x_7	$Cdc20$	$f_7 = CycB$
x_8	$Cdh1$	$f_8 = (\overline{CycA} \wedge \overline{CycB}) \vee Cdc20 \vee (p27 \wedge \overline{CycB})$
x_9	$UbcH10$	$f_9 = \overline{Cdh1} \vee (Cdh1 \wedge UbcH10 \wedge (Cdc20 \vee CycA \vee CycB))$
x_{10}	$CycB$	$f_{10} = (\overline{Cdc20} \wedge \overline{Cdh1})$

corresponding to $f_1 = 0$ and $f_1 = 1$ in Table 1. If $f_1 = 0$, then the healthy and mutated networks have the singleton attractor cycles $\mathcal{A}_0 = \{389\}$ and $\mathcal{A}_1 = \{261\}$, respectively. If $f_1 = 1$, both the healthy and mutated networks have the same attractor cycle $\mathcal{A}_0 = \mathcal{A}_1 = \{516, 524, 527, 583, 613, 629, 561\}$. Consequently, the multiple-cell averaging method cannot classify the two networks when $f_1 = 1$. In the simulations, we assume that the trajectory expression data are generated from the networks with $f_1 = 0$ in both the healthy and mutated networks.

Figs. 8a and 8b show the classification error of the trajectories of length $m = 6$ with $p = 0.05$ and two values of $p_{miss} = 0, 0.2$ versus D for low-noise ($\sigma = 5$) and high-noise ($\sigma = 20$) scenarios, respectively. A higher probability of missing data deteriorates classifier performance, as does higher observation noise σ. Convergence of the error curves is faster for lower σ. The classification error of the healthy and mutated mammalian cell-cycle networks when using averaged expression data in the multiple-cell scenario is shown in Figs. 8c and 8d for $\sigma = 5, 10, 20$, when $p = 0.01$ and $p = 0.05$, respectively. In Figs. 8c and 8d, the classifier based on the multiple-cell expression data can only work well in the low-noise scenarios and is very susceptible to observation noise. Convergence of the error curves versus D gets slower as σ increases. For a given σ, the error of the multiple-cell classifier decreases with decreasing p, the reason being that larger p makes the steady-state distributions of the healthy and mutated cell-cycle BNs similar to each other, so that the means of two normal distributions for the two classes get closer to each other, leading to a larger error.

Figs. 8e and 8f present the error versus m in the mammalian cell-cycle networks for $p = 0.05$, $D = 40$ and different values of $p_{miss} = 0, 0.2, 0.5$ when $\sigma = 5$ and $\sigma = 20$, respectively. Similar to the synthetic networks, the error curves have a decreasing trend as m increases. In Figs. 8e and 8f, the error of the multiple-cell classifier is shown for the same parameter values. In the low-noise scenario of Fig. 8e, the error of the multiple-cell classifier is better than that of the trajectory classifier when m is small, the extent depending on the probability of missing data in the trajectory data; however, for longer trajectories (larger m), the error of the trajectory classifier is less. We observe in Fig. 8f that the performance of the multiple-cell classifier is very bad in the high-noise scenario, the error of the multiple-cell classifier

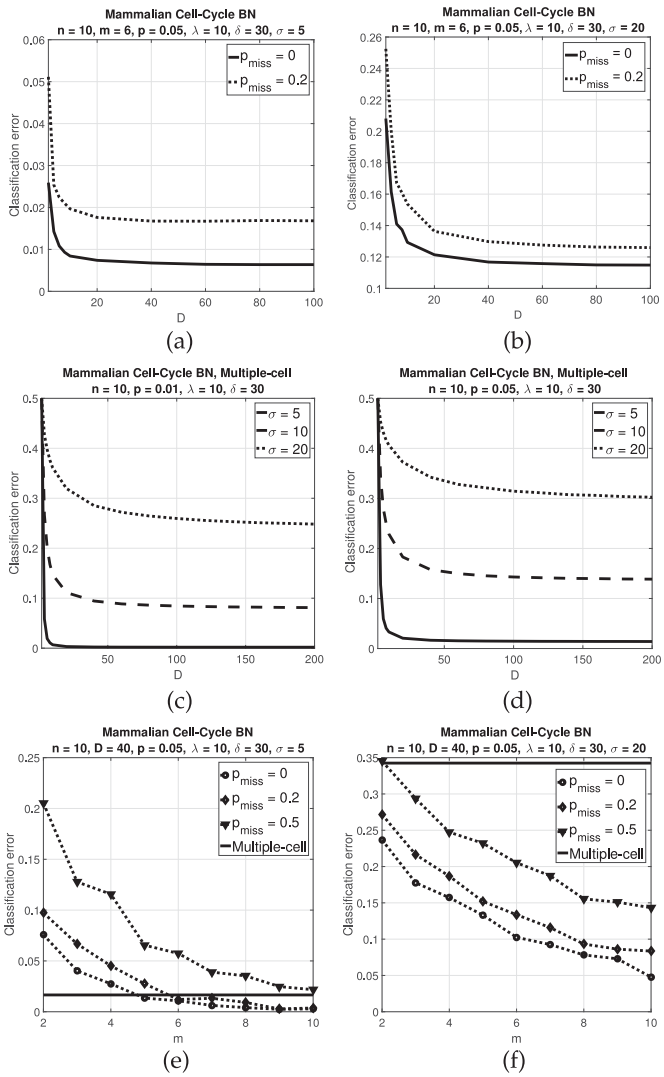

Fig. 8. Classification errors of the trajectory and multiple-cell classifiers in the mammalian cell-cycle BN. The fixed parameters are $n = 10$, $\lambda = 10$, $\delta = 30$. (a) Classification error of the trajectory classifier versus D. The parameters are $m = 6$, $p = 0.05$, $\sigma = 5$. (b) Classification error of the trajectory classifier versus D. The parameters are $m = 6$, $p = 0.05$, $\sigma = 20$. (c) Classification error of the multiple-cell classifier versus D. The parameter is $p = 0.01$. (d) Classification error of the multiple-cell classifier versus D. The parameter is $p = 0.05$. (e) Classification error of the trajectory and multiple-cell classifiers versus m. The parameters are $D = 40$, $p = 0.05$, $\sigma = 5$. (f) Classification error of the trajectory and multiple-cell classifiers versus m. The parameters are $D = 40$, $p = 0.05$, $\sigma = 20$.

being much greater than that of the trajectory classifier for every value of m and even for high p_{miss}.

6 Conclusion

This paper has studied classification of gene-expression trajectories coming from two classes, healthy and mutated (cancerous) using Boolean networks with perturbation (BNps) to model the dynamics of each class at the state level, meaning that each class has its own BNp, which we partially know based on gene pathways. We employ a Gaussian model at the observation level to show the expression values of the genes given the hidden states at each time point. We use the expectation maximization methodology to learn the BNps and the unknown model parameters, derive closed-form updates for the parameters, and propose a learning algorithm. After learning, a plug-in Bayes classifier is used to classify the unlabeled trajectories. The effect of missing data has been considered.

From the biological perspective, measuring gene expressions at different times yields trajectories only when the measurements come from a single cell. In multiple-cell scenarios, the expression values of the genes are averages over many cells with possibly different states. Consequently, using the central-limit theorem, we have proposed another model for expression data in multiple-cell scenarios. Using simulations, it has been demonstrated that single-cell trajectory data can outperform multiple-cell average expression data in terms of the classification error, especially in high-noise situations.

Acknowledgments

The authors acknowledge the support of the National Science Foundation, through NSF award CCF-1320884.

References

[1] A. Karbalayghareh, U. Braga-Neto, J. Hua, and E. R. Dougherty, "Classification of state trajectories in gene regulatory networks," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. PP, no. 99, p. 1, 2017, doi: 10.1109/TCBB.2016.2616470.

[2] M. Imani and U. M. Braga-Neto, "Maximum-likelihood adaptive filter for partially observed boolean dynamical systems," *IEEE Trans. Signal Process.*, vol. 65, no. 2, pp. 359–371, Jan. 2017.

[3] E. R. Dougherty, M. Brun, J. M. Trent, and M. L. Bittner, "Conditioning-based modeling of contextual genomic regulation," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 6, no. 2, pp. 310–320, Apr. 2009.

[4] A. Subramanian, et al., "Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles," *Proc. Nat. Acad. Sci.*, vol. 102, no. 43, pp. 15 545–15 550, 2005.

[5] L. Tian, S. A. Greenberg, S. W. Kong, J. Altschuler, I. S. Kohane, and P. J. Park, "Discovering statistically significant pathways in expression profiling studies," *Proc. Nat. Acad. Sci.*, vol. 102, no. 38, pp. 13 544–13 549, 2005.

[6] A. H. Bild, et al., "Oncogenic pathway signatures in human cancers as a guide to targeted therapies," *Nature*, vol. 439, no. 7074, pp. 353–357, 2006.

[7] M. Ashburner, et al., "Gene ontology: Tool for the unification of biology," *Nature Genetics*, vol. 25, no. 1, pp. 25–29, 2000.

[8] Z. Guo, et al., "Towards precise classification of cancers based on robust gene functional expression profiles," *BMC Bioinf.*, vol. 6, no. 1, 2005, Art. no. 1.

[9] E. Lee, H.-Y. Chuang, J.-W. Kim, T. Ideker, and D. Lee, "Inferring pathway activity toward precise disease classification," *PLoS Comput. Biol.*, vol. 4, no. 11, 2008, Art. no. e1000217.

[10] J. Su, B.-J. Yoon, and E. R. Dougherty, "Accurate and reliable cancer classification based on probabilistic inference of pathway activity," *PloS One*, vol. 4, no. 12, 2009, Art. no. e8161.

[11] C. Trapnell, "Defining cell types and states with single-cell genomics," *Genome Res.*, vol. 25, no. 10, pp. 1491–1498, 2015.

[12] D. Usoskin, et al., "Unbiased classification of sensory neuron types by large-scale single-cell rna sequencing," *Nature Neurosci.*, vol. 18, no. 1, pp. 145–153, 2015.

[13] T. Ilicic, et al., "Classification of low quality cells from single-cell RNA-seq data," *Genome Biol.*, vol. 17, no. 1, Feb. 2016, Art. no. 29.

[14] K. Shekhar, et al., "Comprehensive classification of retinal bipolar neurons by single-cell transcriptomics," *Cell*, vol. 166, no. 5, pp. 1308–1323, 2016.

[15] C. Shao and T. Hfer, "Robust classification of single-cell transcriptome data by nonnegative matrix factorization," *Bioinformatics*, vol. 33, no. 2, pp. 235–242, 2017.

[16] M. Zamanighomi, Z. Lin, T. Daley, A. Schep, W. J. Greenleaf, and W. H. Wong, "Unsupervised clustering and epigenetic classification of single cells," *bioRxiv*, 2017, doi: 10.1101/143701.

[17] Z. Bar-Joseph, et al., "Genome-wide transcriptional analysis of the human cell cycle identifies genes differentially regulated in normal and cancer cells," *Proc. Nat. Acad. Sci.*, vol. 105, no. 3, pp. 955–960, 2008.

[18] A. R. Wu, et al., "Quantitative assessment of single-cell RNA-sequencing methods," *Nature Methods*, vol. 11, no. 1, pp. 41–46, 2014.

[19] A. Ståhlberg and M. Kubista, "The workflow of single-cell expression profiling using quantitative real-time PCR," *Expert Rev. Molecular Diagnostics*, vol. 14, no. 3, pp. 323–331, 2014.

[20] C. Trapnell, et al., "The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells," *Nature Biotechnol.*, vol. 32, no. 4, pp. 381–386, 2014.

[21] P. V. Kharchenko, L. Silberstein, and D. T. Scadden, "Bayesian approach to single-cell differential expression analysis," *Nature Methods*, vol. 11, no. 7, pp. 740–742, 2014.

[22] Z. Wang, et al., "DTWscore: Differential expression and cell clustering analysis for time-series single-cell RNA-seq data," *BMC Bioinf.*, vol. 18, no. 1, May 2017, Art. no. 270.

[23] S. Z. Dadaneh, X. Qian, and M. Zhou, "BNP-seq: Bayesian nonparametric differential expression analysis of sequencing count data," *J. Amer. Stat. Assoc.*, doi: 10.1080/01621459.2017.1328358.

[24] J. E. Reid and L. Wernisch, "Pseudotime estimation: Deconfounding single cell time series," *Bioinformatics*, vol. 32, no. 19, pp. 2973–2980, 2016.

[25] V. Moignard, et al., "Decoding the regulatory network of early blood development from single-cell gene expression measurements," *Nature Biotechnol.*, vol. 33, no. 3, pp. 269–276, 2015.

[26] A. Karbalayghareh, U. Braga-Neto, and E. R. Dougherty, "Classification of gaussian trajectories with missing data in boolean gene regulatory networks," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2017, pp. 1078–1082.

[27] S. A. Kauffman, "Metabolic stability and epigenesis in randomly constructed genetic nets," *J. Theoretical Biol.*, vol. 22, no. 3, pp. 437–467, 1969.

[28] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[29] U. Neto and E. Dougherty, *Error Estimation for Pattern Recognition*. Hoboken, NJ, USA: Wiley, 2015.

[30] E. Shapiro, T. Biezuner, and S. Linnarsson, "Single-cell sequencing-based technologies will revolutionize whole-organism science," *Nature Rev. Genetics*, vol. 14, no. 9, 2013, Art. no. 618.

[31] M. Brun, E. R. Dougherty, and I. Shmulevich, "Steady-state probabilities for attractors in probabilistic boolean networks," *Signal Process.*, vol. 85, no. 10, pp. 1993–2013, Oct. 2005.

[32] A. Fauré, A. Naldi, C. Chaouiya, and D. Thieffry, "Dynamical analysis of a generic boolean model for the control of the mammalian cell cycle," *Bioinformatics*, vol. 22, no. 14, pp. e124–e131, 2006.

Alireza Karbalayghareh received the BSc (hons.) degree from the Iran University of Science and Technology, Tehran, Iran, and the MSc degree from the Sharif University of Technology, Tehran, Iran, in 2011 and 2013, both in electrical engineering. Since 2014, he has been working toward the PhD degree in the Genomic Signal Processing Laboratory, the Department of Electrical and Computer Engineering of Texas A&M University, College Station, TX. His current research interests include genomic signal processing, classification, pattern recognition, and statistical machine learning.

Ulisses Braga-Neto received the PhD degree in electrical and computer engineering from the Johns Hopkins University, Baltimore, MD and the post-doctoral degree from the University of Texas M.D. Anderson Cancer Center, Houston, TX and with the Oswaldo Cruz Foundation, Recife, Brazil. He is an Associate Professor in the Department of Electrical and Computer Engineering and a member of the Center for Bioinformatics and Genomic Systems Engineering, Texas A&M University, College Station, TX. His research interests include pattern recognition and genomic signal processing. He is author of the textbook "Error Estimation for Pattern Recognition" (IEEE-Wiley, 2015) and has received the NSF CAREER Award for his work in this area. He is a Senior Member of the IEEE.

Edward R. Dougherty received the MSc degree in computer science from the Stevens Institute of Technology, and the PhD degree in mathematics from Rutgers University. He has been awarded the Doctor Honoris Causa by the Tampere University of Technology, Finland. He is a Distinguished Professor in the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX, where he holds the Robert M. Kennedy '26 Chair in electrical engineering and is scientific director of the Center for Bioinformatics and Genomic Systems Engineering. He is the author of 16 books, the editor of 5 others, and author of more than 300 journal papers. He is a Fellow of SPIE, has received the SPIE President's Award, and served as the editor of the SPIE/IS&T Journal of Electronic Imaging. At Texas A&M University, he received the Association of Former Students Distinguished Achievement Award in Research, and was named Fellow of the Texas Engineering Experiment Station, and Halliburton Professor of the Dwight Look College of Engineering. He is a Fellow of the IEEE.