PALLAS: Penalized mAximum LikeLihood and pArticle Swarms for Inference of Gene Regulatory Networks from Time Series Data

Yukun Tan[1], Fernando B. Lima Neto[2] and Ulisses Braga Neto[1,*]

[1]Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX, 77843, USA

[2]Department of Computer Engineering, Polytechnic University of Pernambuco, Recife, 50720, Brazil

Abstract

We present PALLAS, a practical method for gene regulatory network (GRN) inference from time series data, which employs penalized maximum likelihood and particle swarms for optimization. PALLAS is based on the Partially-Observable Boolean Dynamical System (POBDS) model and thus does not require ad-hoc binarization of the data. The penalty in the likelihood is a LASSO regularization term, which encourages the resulting network to be sparse. PALLAS is able to scale to large networks under no prior knowledge, by virtue of a novel continuous-discrete Fish School Search particle swarm algorithm for efficient simultaneous maximization of the penalized likelihood over the discrete space of networks and the continuous space of observational parameters. The performance of PALLAS is demonstrated by a comprehensive set of experiments using synthetic data generated from real and artificial networks, as well as real time series microarray and RNA-seq data, where it is compared to several other well-known methods for gene regulatory network inference. The results show that PALLAS can infer GRNs efficiently and accurately. PALLAS is a fully-fledged program with a command-line user interface, written in python, and available on GitHub (`https://github.com/yukuntan92/PALLAS`).

1 Introduction

Inference of gene regulatory networks (GRN) from gene expression time-series data is a problem of critical importance in Bioinformatics [12]. Many mathematical models have been proposed in the literature to address this problem, including linear models [48, 13], Bayesian networks [37, 18], neural networks [49], differential equations [13, 10] and information theory based approaches [15, 4]. The Boolean network (BN) model [29], is an effective model for GRNs due to its ability to describe temporal patterns of gene activation and inactivation and its comparatively small data requirement for inference [1, 33, 16, 24, 43]. Several extensions of the BN model have been proposed, including Random Boolean Networks [29], Boolean Networks with perturbation (BNp) [41], and Probabilistic Boolean Networks (PBN) [42], and Boolean Control Networks (BCN) [11, 38]. However, all of those models assume that the system Boolean states are completely observable. This is a significant drawback, since all practical methods for the inference of Boolean networks must include a step of ad-hoc binarization of the gene expression data. The Partially-observed Boolean dynamical system (POBDS) model [9] addresses this problem in a principled way, by postulating separate Boolean state and general observation processes. The time-series gene expression data, whether microarray or RNA-seq data, is modeled by the observation process, while the Boolean states are hidden. This allows the optimal inference of the sequence of Boolean states from the time series data, as well as parameter estimation directly from the data.

In this paper, we present PALLAS, a practical method for parametric gene network inference based on the POBDS model, using penalized maximum likelihood and particle swarms for optimization. The penalty in the likelihood score is a L_1-norm LASSO regularization term [47], which encourages the resulting network to be sparse, i.e., contain a small number of edges between genes; its value can be adjusted by the user to obtain a desired level of sparsity. The likelihood itself is calculated efficiently by an auxiliary particle filter (APF) implementation of the Boolean Kalman Filter [9, 25]. Another novel feature of PALLAS is the application to Boolean models of a particle swarm method: a new mixed continuous-discrete version of the Fish School Search algorithm [6, 7], for efficient simultaneous maximization of the penalized likelihood over the discrete space of networks and the continuous space of observational parameters. An early version of this work appeared previously in a short communication [46]. We mention that particle swarm methods have been employed before for GRN inference using non-Boolean models, namely, recursive neural networks (RNN) in [50] and S-systems in [28].

PALLAS is an extension of the adaptive filtering method proposed in [25]. The latter performs maximization of the likelihood function by exhaustive search over the space of networks and expectation maximization over the space of parameters of the observational model for each candidate network. It is well suited if there is prior knowledge about the network, e.g., most of the edges are known and only a few putative edges are being sought, given the prohibitive computational cost of exhaustively searching the space of all networks. For example, with only 4 genes, there are a total of 688,747,536 Boolean network models to be searched,

and with 8 genes this number jumps to approximately 8.8×10^{32}, rendering exhaustive search completely unfeasible. PALLAS differs from the method in [25] in using penalized maximum likelihood and particle swarms for optimization, which allows it to handle large networks in the absence of any prior knowledge.

The performance of PALLAS is demonstrated by a comprehensive set of experiments. Using synthetic data generated from both real and artificial GRNs, which allows computation of performance metrics, we compare PALLAS to GENIE3 [27], TIGRESS [21], Banjo [44], Best-Fit algorithm [32], REVEAL [34], and GABNI [5]. Using real time series microarray data from the SOS DNA Repair System in E. Coli, we compare PALLAS to the methods in [31, 30, 28]. We also illustrate the performance of PALLAS in recovering known regulatory links in the E. Coli Biofilm Formation Pathway using time series RNA-Seq data.

2 Methods

2.1 Partially-Observable Boolean Dynamical Systems

The Partially-Observable Boolean dynamical system (POBDS) model [9] allows for uncertainty in Boolean state transitions and partial observation of the Boolean state variables through noise.

2.1.1 State model

Consider a state process $\{\mathbf{X}_k; k = 0, 1, \ldots\}$, where $\mathbf{X}_k \in \{0,1\}^d$ is a Boolean vector of size d, which evolves according to

$$\mathbf{X}_k \;=\; \mathbf{f}\left(\mathbf{X}_{k-1}\right) \oplus \mathbf{n}_k\,, \tag{1}$$

for $k = 1, 2, \ldots$ where $\mathbf{f} : \{0,1\}^d \longrightarrow \{0,1\}^d$ is called the *network function*, $\mathbf{n}_k \in \{0,1\}^d$ is additive noise at time k, and "$\oplus$" indicates component-wise modulo-2 addition. The state and noise processes are assumed to be independent. The state model (1) can be suitably modified to include external inputs, if desired.

The noise random vector $\mathbf{n}_k$ models uncertainty in the state transition: if a component of $\mathbf{n}_k$ is 1, the corresponding component of $\mathbf{f}\left(\mathbf{X}_{k-1}\right)$ is flipped. As long as all components of $\mathbf{n}_k$ have a nonzero probability of being 1, the state process is an ergodic Markov Chain, with a steady state distribution. But if the noise is too intense, i.e., the probability of 1's in $\mathbf{n}_k$ is too large, state evolution becomes chaotic. However, it is well known that important biological pathways are tightly regulated. Accordingly, each component of the noise vector is assumed here to be equal to 1 with a small value $p = 0.05$, independently of the others. A different value $0 \leq p \leq 0.5$ can be selected by the user; however, a value much larger or smaller than 0.05 is not representative of real gene regulatory networks.

We assume a specific model for the network function that is motivated by gene pathway diagrams commonly encountered in biomedical research. Let a sample state vector $\mathbf{x} \in \{0,1\}^d$ and the network function $\mathbf{f}$ be expressed in component form as $\mathbf{x} = (x_1, \ldots, x_d)$ and $\mathbf{f} = (f_1, \ldots, f_d)$, respectively. Each component $f_i : \{0,1\}^d \to \{0,1\}$ is given by

$$f_i(\mathbf{x}) \;=\; \begin{cases} 1, & \sum_{j=1}^{d} a_{ij} x_j + b_i > 0\,, \\ 0, & \text{otherwise}, \end{cases} \tag{2}$$

where $a_{ij} = +1$ if there is positive regulation (activation) from gene j to gene i; $a_{ij} = -1$ if there is negative regulation (inhibition) from gene j to gene i; and $a_{ij} = 0$ if gene j is not an input to gene i, whereas $b_i = +1/2$ if gene i is positively biased in the sense that an equal number of activation and inhibition inputs will produce activation; the reverse being the case if $b_i = -1/2$. The network model is depicted in Figure 1, where the threshold units are step functions that output 1 if the input is positive, and 0, otherwise. This model constraint reduces the number of parameters needed to specify $\mathbf{f}$ from 2^d to $d^2 + d$.

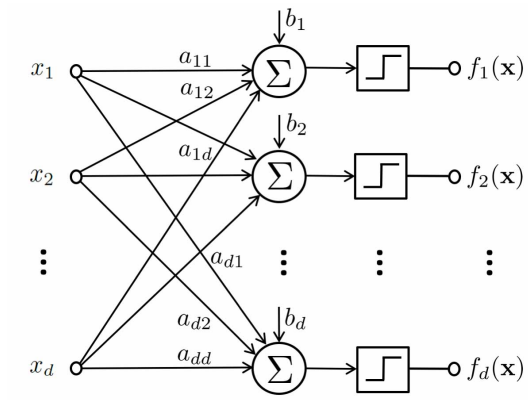

Figure 1: Schematic representation of the network function.

2.1.2 Observation Model

The sequence of states is observed indirectly through the process $\{\mathbf{Y}_k; k = 0, 1, \ldots\}$, where the measurement vector $\mathbf{Y}_k$ is a general nonlinear function of the state and observation noise:

$$\mathbf{Y}_k \;=\; \mathbf{h}\left(\mathbf{X}_k, \mathbf{v}_k\right) \tag{3}$$

for $k = 1, 2, \ldots$, where the noise vector $\mathbf{v}_k$ is assumed to independent of the state process and state transition noise process. We describe next the two observational models considered in this paper, corresponding to two common gene expression modalities: RNA-Seq count data or microarray fluorescence data. Observational models for other data modalities can be introduced, if desired.

<u>RNA-Seq observation model</u>

RNA-Seq data can be modeled with the Poisson distribution [19] or the negative binomial distribution [20, 2]. Here, we employ the latter, since it is able to address overdispersion in the count distributions. We assume that the transcript counts $\mathbf{Y}_k = (Y_{k1}, \ldots, Y_{kd})$ are re-

lated to the state $\mathbf{X}_k = (X_{k1}, \ldots, X_{kd})$ via

$$P(\mathbf{Y}_k = \mathbf{y} \mid \mathbf{X}_k = \mathbf{x}) = \prod_{i=1}^{d} P(Y_{ki} = y_i \mid X_{ki} = x_i), \tag{4}$$

and adopt the negative binomial model for each count,

$$P(Y_{ki} = y_i \mid X_{ki} = x_i) = \frac{\Gamma(y_i + \phi_i)}{y_i!\,\Gamma(\phi_i)} \left(\frac{\lambda_i}{\lambda_i + \phi_i}\right)^{y_i} \left(\frac{\phi_i}{\lambda_i + \phi_i}\right)^{\phi_i}, \tag{5}$$

where Γ denotes the Gamma function, and $\phi_i, \lambda_i > 0$ are the real-valued inverse dispersion parameter and mean read count of transcript i, respectively, for $i = 1, \ldots, d$. The inverse dispersion parameter ϕ_i specifies the amount of observation noise: the larger it is, the less observation noise is present. We model the parameter λ_i in log-space as:

$$\log \lambda_i = \log s + \mu_i + \delta_i x_i, \tag{6}$$

where the parameter s is the sequencing depth, which depends on the instrument, $\mu_i \geq 0$ is the baseline level of expression in the inactivated transcriptional state, and $\delta_i > 0$ is the difference between read count as gene i goes from the inactivated ($x_i = 0$) to the activated ($x_i = 1$) state, for $i = 1, \ldots, d$.

Microarray observation model

A reasonable model for continuous microarray fluorescence data is a Gaussian linear model:

$$\mathbf{y} = \boldsymbol{\mu} + D\,\mathbf{x} + \mathbf{v}, \tag{7}$$

where $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_d) \geq 0$ is the vector of baseline expression levels corresponding to the "zero" or inactive state for each gene, $D = \mathrm{diag}\{\delta_1, \ldots, \delta_d\} > 0$ is a diagonal matrix containing differential expression values for each gene, and $\mathbf{v} \sim \mathcal{N}(0, \Sigma)$ is an uncorrelated zero-mean Gaussian noise vector, where $\Sigma = \mathrm{diag}\{\sigma_1^2, \ldots, \sigma_d^2\} > 0$. Notice that (4) is still satisfied here.

2.1.3 Boolean Kalman Filter

Given a time series of observations $\mathbf{Y}_{1:k} = \{\mathbf{Y}_1, \ldots, \mathbf{Y}_k\}$, the Boolean Kalman Filter[9] (described in detail in the Supplementary Material) computes exactly the minimum mean-square error state estimator:

$$\hat{\mathbf{X}}_k^{\mathrm{MS}} = \underset{\hat{\mathbf{X}}_k \in \{0,1\}^d}{\mathrm{argmin}} \; E[\,||\hat{\mathbf{X}}_k - \mathbf{X}_k||^2 \mid \mathbf{Y}_{1:k}]\,. \tag{8}$$

The BKF also computes the probabilities needed to determine the likelohood function, as detailed in Section 2.2.1. When the network is large, however, computation of the BKF is intractable since each transition matrix contain 2^{2d} elements which requires large computation and memory. In this case, approximate methods must be used, such as the Sequential Monte Carlo (SMC) method, also known as particle filter [3]. Here we use the auxiliary particle filter implementation of the Boolean Kalman Filter (APF-BKF), described in [26] (please see that reference for the details).

2.2 PALLAS Algorithm

In this section, we describe in detail PALLAS (Penalized mAximum LikeLihood and pArticle Swarms), an algorithm for inference of Boolean gene regulatory networks from noisy time series of gene expression data. The algorithm has two main components: 1) efficient computation of a penalized log-likelihood cost function; 2) maximization of the previous cost function using a novel particle swarm method, namely, a mixed discrete-continuous fish school search procedure. We describe here the general case, where no prior knowledge is available to set model parameters. The algorithm can be easily modified to allow some of the parameters to be specified by the user, by simply reducing the size of the parameter space and using the known parameter values in the likelihood computation.

Let $\theta = (\theta_{\mathrm{disc}}, \theta_{\mathrm{cont}}) \in \Theta$, with $\theta_{\mathrm{disc}} \in \Theta_{\mathrm{disc}}$ and $\theta_{\mathrm{cont}} \in \Theta_{\mathrm{cont}}$, be the discrete and continuous unknown model parameters, where Θ, Θ_{disc} and Θ_{cont} are the corresponding parameter spaces, with $\Theta = \Theta_{\mathrm{disc}} \times \Theta_{\mathrm{cont}}$. Here, θ_{disc} contains the parameters of the network function in (2), namely the edge parameters $a_{ij} \in \{-1, 0, 1\}$, for $i, j = 1, \ldots, d$, and the regulation bias parameters $b_i \in \{-1/2, 1/2\}$, for $i = 1, \ldots, d$. Hence, $\Theta_{\mathrm{disc}} = \{-1, 0, 1\}^{d^2} \times \{-1/2, 1/2\}^d$. This is a finite space, but its cardinality $|\Theta_{\mathrm{disc}}| = 3^{d^2} \times 2^d$ increases extremely fast with the number of genes d. For example, for a network with only $d = 4$ genes, $|\Theta_{\mathrm{disc}}| = 688747536$, while if $d = 8$, then $|\Theta_{\mathrm{disc}}| \approx 8.8 \times 10^{32}$. On the other hand, θ_{cont} contains the observational parameters: the baseline expression levels $\mu_i > 0$ and the differential expression levels $\delta_i > 0$, for $i = 1, \ldots, d$, for both RNA-Seq and microarray data, the inverse dispersion parameters $\phi_i > 0$, for $i = 1, \ldots, d$, for RNA-Seq data, and the standard deviations $\sigma_i > 0$, for $i = 1, \ldots, d$, for microarray data (the sequencing depth parameter s is assumed known for a given RNA-seq assay, so it is not part of θ_{cont}). Hence, the dimensionality of θ_{cont} is $Q = 3d$ in both cases.

The mixed discrete-continuous fish school search procedure employed by PALLAS assumes that the parameter space is a closed and bounded region with an absorbing decision boundary (if the current best estimate exceeds the boundary, it remains at the boundary). This is not a limiting requirement in practice, since sensible lower and upper bounds can be set for all the observational parameters. These intervals can be set by the user, or the following data-driven procedure to obtain default intervals is employed. Let min, max, and mean be respectively the minimum, maximum, and mean value of the observed data for all genes across all time points and available time series. In the case of RNA-Seq data, the data must be normalized by dividing the measurements by the sequencing depth and then taking logs prior to computing the mean, max, and mean values. Then the following intervals are assumed:

$$\begin{aligned}
\mu_i &\in [\,\mathrm{min}\,,\,\mathrm{mean}\,]\,,\\
\delta_i &\in [\,\mathrm{min}\{\mathrm{max} - \mathrm{mean}, \mathrm{mean} - \mathrm{min}\}/3\,,\,\mathrm{max} - \mathrm{min}\,]\,,\\
\sigma_i &\in [\,0.1\,,\,\mathrm{max}\{\mathrm{max} - \mathrm{mean}, \mathrm{mean} - \mathrm{min}\}/3\,]\,,\\
\phi_i &\in [\,0.5, 7\,]\,,
\end{aligned} \tag{9}$$

for $i = 1, \ldots, d$. Some of the parameters can often be assumed to be the same across different genes, which reduces the data requirement of the estimation problem. In the simplest case, there are single parameters μ, δ, and σ or ϕ for all genes, so that $Q = 3$. In any case, Θ_{cont} is a closed and bounded rectangular region in R_+^Q.

Next we describe the two main steps comprising the PALLAS algorithm: computation of the penalized log-likelihood function and the novel mixed discrete-continuous fish-school search method.

2.2.1 Penalized Maximum-Likelihood Computation

Suppose that the sample data consist of n independent time series $\mathbf{Y}_{1:k}^j = \{\mathbf{Y}_1^j, \ldots, \mathbf{Y}_k^j\}$ up to time k, for $j = 1, \ldots, n$. The penalized log-likelihood of model θ at time k is defined as

$$\begin{aligned} L_k(\theta) &= \frac{1}{kn} \log p_\theta(\mathbf{Y}_{1:k}^{(1)}, \ldots, \mathbf{Y}_{1:k}^{(n)}) - \eta \sum_{i,j=1}^{2^d} |a_{ij}| \\ &= \frac{1}{kn} \sum_{j=1}^{n} \log p_\theta(\mathbf{Y}_{1:k}^j) - \eta \sum_{i,j=1}^{2^d} |a_{ij}| \,, \end{aligned} \tag{10}$$

where $\eta > 0$ is a regularization parameter, which has a default value of $\eta = 0.01$ in our implementation. Hence, the penalized log-likelihood in (10) is the sum of the average log-likelihood per time series and a negative value times the number of edges in the model. Maximization of (10) thus encourages the model to both fit the data and be sparse, i.e., contain a small number of edges between genes, which is in agreement with biological knowledge. The value of η can be adjusted by the user to obtain a desired level of sparsity. Notice that

$$\begin{aligned} \log p_\theta(\mathbf{Y}_{1:k}^j) &= \log \Big[p_\theta(\mathbf{Y}_k^j | \mathbf{Y}_{1:k-1}^j) p_\theta(\mathbf{Y}_{k-1}^j | \mathbf{Y}_{1:k-2}^j) \\ &\qquad \cdots p(\mathbf{Y}_2^j \mid \mathbf{Y}_1^j) p(\mathbf{Y}_1^j) \Big] \\ &= \sum_{m=1}^{k} \log p_\theta(\mathbf{Y}_m^j \mid \mathbf{Y}_{1:m-1}^j) \,, \end{aligned} \tag{11}$$

where

$$\begin{aligned} & p_\theta(\mathbf{Y}_m^j \mid \mathbf{Y}_{1:m-1}^j) \\ &= \sum_{i=1}^{2^d} p_\theta(\mathbf{Y}_m^j | \mathbf{X}_m = \mathbf{x}^i, \mathbf{Y}_{1:m-1}^j) P_\theta(\mathbf{X}_m = \mathbf{x}^i | \mathbf{Y}_{1:m-1}^j) \\ &= \sum_{i=1}^{2^d} p_\theta(\mathbf{Y}_m^j \mid \mathbf{X}_m = \mathbf{x}^i) \; P_\theta(\mathbf{X}_m = \mathbf{x}^i \mid \mathbf{Y}_{1:m-1}^j) \,. \end{aligned} \tag{12}$$

With $(\beta_m^{\theta,j})_i = p_\theta(\mathbf{Y}_m^j \mid \mathbf{X}_m = \mathbf{x}^i) \; P_\theta(\mathbf{X}_m = \mathbf{x}^i \mid \mathbf{Y}_{1:m-1}^j)$, the penalized log-likelihood in (10) be written as

$$L_k(\theta) = \frac{1}{kn} \sum_{j=1}^{n} \sum_{m=1}^{k} ||\beta_m^{\theta,j}||_1 - \eta \sum_{i,j=1}^{2^d} |a_{ij}| \,. \tag{13}$$

The sequence of values $||\beta_m^{\theta,j}||_1$, for $j = 1, \ldots, n$ and $m = 1, \ldots, k$, can be computed by a BKF tuned to parameter θ applied to the time series $\mathbf{Y}_{1:k}^j$ (see the Supplementary Material for a description of the BKF). As mentioned in the previous section, here we use the auxiliary particle filtering implementation of the BKF, for computational efficiency. The maximum-likelihood estimator of parameter θ at time k is then given by

$$\hat{\theta}_k^{\text{ML}} = \arg\max_{\theta \in \Theta} L_k(\theta) \,. \tag{14}$$

A state estimate $\hat{\mathbf{X}}_k^{\text{ML}} = \hat{\mathbf{X}}_k(\hat{\theta}_k^{\text{ML}})$ can be obtained, if desired, where $\hat{\mathbf{X}}_k(\theta)$ denotes the optimal state estimator produced by a BKF tuned to the parameter θ.

2.2.2 Mixed Fish School Search Algorithm

In this section, we describe in detail a novel particle-swarm algorithm for optimization over a combined discrete-continuous parameter space, called the mixed fish school search (MFSS) algorithm, which is an extension of the fish school search (FSS) algorithm for continuous parameter spaces proposed in [6].

In the MFSS algorithm, the objective is to find a model that maximizes a given score or fitness — in our present case, this is the penalized log-likelihood defined in the previous section. Each candidate model, i.e., each candidate parameter vector $\theta = (\theta_{\text{disc}}, \theta_{\text{cont}})$, corresponds to a particle or "fish." The length of θ is denoted by P. From the previous section, $P = d^2 + d + Q$. The fish school is an ensemble of S such particles in the parameter space $\Theta = \Theta_{\text{disc}} \times \Theta_{\text{cont}}$. The position of fish s at iteration r will be denoted by $\theta^s(r) = (\theta_{\text{disc}}^s(r), \theta_{\text{cont}}^s(r))$, for $s = 1, \ldots, S$, and $r = 0, \ldots, R$. The number of fishes S and the total number of iterations R are user-defined parameters (in practice, $S = 3 \times P$ and $R = 5000$ are found to be good values). In addition, each fish s has a weight $w_s(r)$ at iteration r, which reflects the quality of the solution.

<u>Initialization</u>

The initial position $\theta^s(0) = (\theta_{\text{disc}}^s(0), \theta_{\text{cont}}^s(0))$ of each fish is assigned randomly. The continuous vector $\theta_{\text{cont}}^s(0)$ is drawn from a uniform distribution over Θ_{cont}, but for the discrete part, it is advantageous to use a non-uniform distribution to initialize the edge parameters, in such a way that $a_{ij}^s(0)$ is equal to -1 or 1 with probability 1/4, and 0 with probability 1/2, for $i, j = 1, \ldots, d$, which introduces a bias towards 0 over 1 and -1. This is in agreement with the biological observation that GRNs tend to be sparsely connected. The initial value $b_i^s(0)$ of the regulation bias parameter is chosen to be either $-1/2$ or $1/2$ with equal probabilities, for $i = 1, \ldots, d$.

<u>Individual movement operator</u>

This is an exploratory step, where each fish independently moves a short distance in a random direction, as long as this increases the fitness function. Let $\Delta\theta_{\text{ind}}^s(r) = (\Delta\theta_{\text{disc,ind}}^s(r), \Delta\theta_{\text{cont,ind}}^s(r))$ be the (candidate) individual displacement vector for fish s at iteration r. Vector $\Delta\theta_{\text{disc,ind}}^s(r)$ is drawn from a uniform distribution over the rectangular region $[-1, 1]^{d^2+d}$, while $\Delta\theta_{\text{cont,ind}}^s(r)$ is drawn from a uniform distribution over the rectangular region $[-\tau_1(r), \tau_1(r)] \times \cdots \times [-\tau_Q(r), \tau_Q(r)]$. The step size bounds $\tau_q(r)$, for $q = 1, \ldots, Q$, shrink linearly with r, in order to ensure convergence and emphasize exploitation over exploration at

later iterations. In our implementation, the initial and final values $\tau_q(1)$ and $\tau_q(R)$ are set, respectively, to 10% and 0.01% of the range (i.e., the difference between upper and lower bounds) of the corresponding continuous parameter — these values can be modified by the user, if desired. Now, $\Delta\theta^s_{\text{disc,ind}}(r)$ needs to be quantized into the lattice $\{-1,0,1\}^{d^2+d}$ in order to be added to the discrete component of the current fish position. The quantization scheme we adopt here is a generalization of the method for binary parameters in [40]. We define two adaptive thresholds:

$$\begin{aligned} \text{thr}^s_{\text{pos}}(r) &= \max_+(\Delta\theta^s_{\text{disc,ind}}(r)) \times \tfrac{r}{R}\,, \\ \text{thr}^s_{\text{neg}}(r) &= \min_-(\Delta\theta^s_{\text{disc,ind}}(r)) \times \tfrac{r}{R}\,, \end{aligned} \tag{15}$$

where the operator $\max_+(\mathbf{v})$ is equal to the maximum of the components of vector $\mathbf{v}$ if at least one of them is positive, and equal to zero, otherwise; similarly, $\min_-(\mathbf{v})$ is equal to the minimum of the components of $\mathbf{v}$ if at least one of them is negative, and equal to zero, otherwise. The factor r/R increases the thresholds (in magnitude) with r, to favor exploitation over exploration at later iterations and ensure convergence.

The quantized discrete displacement vector is obtained by assigning 1 to a positive component if it is larger than $\text{thr}^s_{\text{pos}}(r)$, assigning -1 to a negative component if it is smaller than $\text{thr}^s_{\text{neg}}(r)$, and assigning 0 to all other components (no movement). Then the position of fish s is updated if the exploratory move causes an increase in fitness:

$$\theta^s_{\text{ind}}(r) = \begin{cases} \theta^s(r-1) + \Delta\theta^s_{\text{ind}}(r), \\ \quad \text{if } L_k(\theta^s(r-1)+\Delta\theta^s_{\text{ind}}(r)) > L_k(\theta^s(r-1)), \\ \theta^s(r-1), \qquad \text{otherwise.} \end{cases} \tag{16}$$

where L_k is the penalized log-likelihood of the model, defined in the previous section. An absorbing boundary condition is adopted, whereby each fish interrupts its movement at the boundary of the parameter space, at the point where it encounters it.

Feeding operator

The weights of all fish are updated based on the fitness improvement from the previous individual movement, if any:

$$w^s(r) = w^s(r-1) + \frac{L_k(\theta^s_{\text{ind}}(r)) - L_k(\theta^s(r-1))}{\max_s\{L_k(\theta^s_{\text{ind}}(r)) - L_k(\theta^s(r-1))\}}\,. \tag{17}$$

Collective instinctive movement operator

This operator makes the fish that had successful individual movements influence the collective direction of movement of the school. The position of each fish s is updated according to:

$$\theta^s_{\text{inst}}(r) = \theta^s_{\text{ind}}(r) + \frac{\sum_{s'=1}^S \Delta\theta^{s'}_{\text{ind}}(r)(L_k(\theta^{s'}_{\text{ind}}(r)) - L_k(\theta^{s'}(r-1)))}{\sum_{s'=1}^S (L_k(\theta^{s'}_{\text{ind}}(r)) - L_k(\theta^{s'}(r-1)))}\,. \tag{18}$$

The displacement in discrete parameter space is quantized following the same procedure adopted to discretize the individual movement displacement vector.

Collective volitive movement operator

This is similar to the individual movement step, but now the fish move in concert, depending on whether the fish school is successful after the previous steps, i.e., its total weight increases, or not. If the fish school is successful, then it should contract, changing from exploration to exploitation mode. Otherwise, it should expand in order to explore the space more. This is accomplished by first defining the current fish school barycenter:

$$\mathbf{b}(r) = \frac{\sum_{s=1}^S w^s(r)\theta^s_{\text{inst}}(r)}{\sum_{s=1}^S w^s(r)}\,. \tag{19}$$

For each fish s, after the collective instinctive movement at iteration r, let $\boldsymbol{\xi}^s(r) = \theta^s_{\text{inst}}(r) - \mathbf{b}(r) = (\xi^s_1(r),\ldots,\xi^s_R(r))$ be the position vector with respect to the school barycenter. Let $\Delta\theta^s_{\text{vol}}(r) = (\Delta\theta^s_{\text{disc,vol}}(r), \Delta\theta^s_{\text{cont,vol}}(r))$ be the collective volitive displacement vector for fish s at iteration r. Vector $\Delta\theta^s_{\text{disc,vol}}(r)$ is drawn from a uniform distribution over the rectangular region $[0,\xi^s_1]\times\cdots\times[0,\xi^s_{d^2+d}]$ and quantized by the same process used in the individual move, while $\Delta\theta^s_{\text{cont,vol}}(r)$ is drawn from uniform distribution over the rectangular region $[0, 2\tau_1(r)\xi^s_{d^2+d+1}(r) \times \cdots \times [0, 2\tau_Q(r)\xi^s_{d^2+d+Q}(r)]$, where $\tau_1(r),\ldots,\tau_Q(r)$ are the same step sizes used in the individual movement step. If the school is successful, i.e., if $\sum_{s=1}^S w^s(r) > \sum_{s=1}^S w^s(r-1)$, then its radius should contract, and

$$\theta^s_{\text{vol}}(r) = \theta^s_{\text{inst}}(r) - \Delta\theta^s_{\text{vol}}(r)\,, \tag{20}$$

otherwise, the radius expands, so the school can escape a bad region, and

$$\theta^s_{\text{vol}}(r) = \theta^s_{\text{inst}}(r) + \Delta\theta^s_{\text{vol}}(r)\,, \tag{21}$$

The result is the new position of the fish $\theta^s(r)$.

3 Results

In this section, we present the result of a comprehensive set of numerical experiments, using both synthetic and real gene expression time series data, to assess the performance of PALLAS and compare it against that of other popular methods in the literature. No prior knowledge is used; i.e., all model parameters must be estimated. Unless otherwise noted, the default values for all PALLAS fixed parameters and estimation intervals are used, as described in the previous sections.

3.1 Performance Criteria

The problem of comparing networks is a nontrivial one; there is not a single metric that captures both the topological and dynamical properties of the networks [14]. Here we consider two classes of metrics, one based on the difference between the network functions, which takes into account the full regulatory relationships among genes, and the other based on edge-calling error rates, which considers only the network topology.

3.1.1 Network Function Distance

Let $\mathbf{f} = (f_1, \ldots, f_d)$ and $\hat{\mathbf{f}} = (\hat{f}_1, \ldots, \hat{f}_d)$ be the network functions of the groundtruth and inferred networks, where the component functions f_i and $\hat{f}_i$ are Boolean functions on d variables, for $i = 1, \ldots, d$; see (1). The performance criterion is the average number of disagreeing Boolean functions between the two networks

$$\varphi(\mathbf{f}, \hat{\mathbf{f}}) = \frac{1}{d \times 2^d} \sum_{i=1}^{d} \sum_{j=1}^{2^d} \left[f_i(\mathbf{x}^j) \oplus \hat{f}_i(\mathbf{x}^j) \right]. \quad (22)$$

This distance is related to the dynamical behavior of the networks, since it has to do with how the Boolean functions differ.

3.1.2 Edge-Calling Error Rates

An *edge* in the groundtruth network represents a relationship between two genes. Here we consider directionality (an edge from gene i to gene j is distinct from an edge from gene j to gene i), but disregard activation/inhibition relationships (this is done because some of the methods to which PALLAS is compared in this section do not capture activation/inhibition). Let TP and FN be the total number of directional edges that are correctly detected (irrespective of inhibition/activation) and incorrectly missed by the inference algorithm, respectively. Similarly, let FP and TN be the total number of directional edges that are incorrectly found and correctly missed, respectively. We define the following edge-calling error rates:

(i) Sensitivity/True Positive Rate (TPR):

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \quad (23)$$

(ii) Specificity/True Negative Rate (SPC):

$$\text{SPC} = \frac{\text{TN}}{\text{FP} + \text{TN}}. \quad (24)$$

(iii) Precision/Positive Predictive Value (PPV):

$$\text{PPV} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \quad (25)$$

3.2 Experiments with Synthetic Data

3.2.1 Mammalian Cell-Cycle Network with Synthetic RNA-Seq Data

Here, we present results based on the well-known Mammalian Cell-Cycle network [17], which is displayed in Figure 2. (Results for a different GRN are presented in the Supplementary Material). The state vector is $\mathbf{X}$ = (CycD, Rb, p27, E2F, CycE, CycA, Cdc20, Cdh1, UbcH10, CycB). This is a large network, with a huge parameter space, for which the estimation problem is hard. The gene interaction parameters a_{ij} can be read from Figure 2 in the same way as in the last section. Once again, the regulation biases are set to $b_i = -1/2$, for $i = 1, \ldots, 10$. The transition noise parameter p is selected randomly in the interval $[0.01, 0.1]$. The RNA-Seq

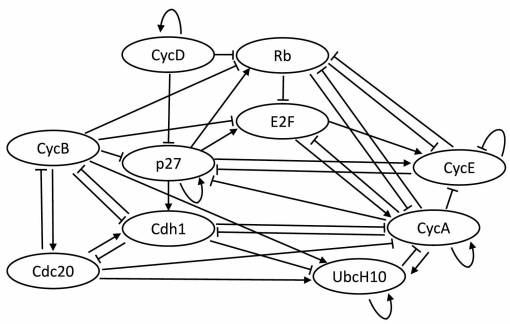

Figure 2: Mammalian cell cycle network.

data model parameters are $\mu_i \equiv \mu = 0.1$, $\delta_i \equiv \delta = 3$, $\phi_i \equiv \phi = 5$, for $i = 1, \ldots, 10$. The sequencing depth is set to $s = 22.52$ (500K-550K reads) and the time series length is fixed at 50. Here we compare PALLAS with the GENIE3 [27], TIGRESS [21], and Banjo [44] algorithms. Like PALLAS, these algorithms can operate directly on the noisy time series, without a need for ad-hoc binarization. However, they do not estimate observational parameters or provide activation/inhibition information, so only the edge-calling error rates in Section 3.1.2 are appropriate here. Average rates obtained over 20 repetitions of the experiment are displayed in Figure 3. One can see that with similar specificity, PALLAS displays higher sensitivity and precision than GENIE3 and TIGRESS. Although it was not possible to adjust the specificity of Banjo to the same levels, we can see that its sensitivity is quite low. In fact, Banjo returned a very small number of edges overall in this experiment. PALLAS also displayed the highest precision among all the algorithms.

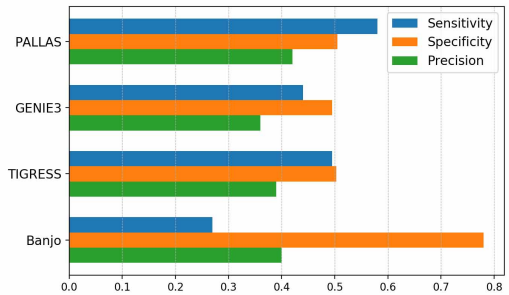

Figure 3: Mammalian cell cycle experiment results.

3.2.2 Artificial Networks with Synthetic RNA-Seq Data

In this section we report results obtained on an ensemble of 10 randomly generated networks with $d = 8$ genes, where each gene is regulated by 3 other genes on average. Edge connectivity, including activation and inhibition, as well as regulation biases, are randomly chosen. The transition noise parameter p is selected randomly in the interval $[0.01, 0.1]$. RNA-Seq synthetic data are generated with parameters $\mu_i \equiv \mu = 0$, $\phi_i \equiv \phi = 1$ or 5, for $i = 1, \ldots, 8$. In the first case, there is more observation noise, and the problem is harder. The parameters

δ_i are allowed to vary uniformly over the intervals $[1, 2]$ or $[1, 5]$, for $i = 1, \ldots, 8$. In the first case, the problem is harder, since the differences in observed expression are smaller. Sequencing depth is set at $s = 22.52$ (500K-550K reads).

Here, we compare PALLAS with the Best-Fit [32], REVEAL [34], and GABNI [5] algorithms. These methods apply to Boolean time series, so they need to employ ad-hoc binarization of the gene expression data. For the first two, [8] recommends the use of the KM3 binarization method, while for GABNI, [4] recommends the use of K-means binarization; hence, we use those binarization methods here. The output of the Best-Fit and REVEAL algorithms are Boolean transition functions, for which the network function distance is appropriate. On the other hand, the output of GABNI consist of positive (activating) or negative (inhibitory) interactions, for which we use the edge-calling error rates defined previously.

Average network function distances and edge-calling error rates obtained over 20 repetitions of the experiment (2 for each of the 10 networks) with $\phi = 5$ are displayed in Figures 4 and 5 (corresponding results for $\phi = 1$ are shown in the Supplementary Material). Figure 4 shows that the performance of Best-Fit and PALLAS increases with the time series length, while the performance of REVEAL is mostly stable. PALLAS perform better than the Best-Fit algorithm, especially when δ is smaller. This reflects the fact that ad-hoc binarization of the data becomes less accurate with a smaller difference between activation/inactivation levels in the observed data, which is determined by δ. Figure 5 shows that PALLAS beats GABNI in sensitivity throughout, as well as in specificity under low observation noise and sufficient data. Indeed, the specificity of GABNI is artificially large for small amounts of data, when it detects very few edges.

3.3 Experiments with Real Data

3.3.1 E. Coli SOS DNA Repair System

In this section, we demonstrate the application of PALLAS to real microarray data from a well-known biological system, namely, the SOS DNA repair system in E. Coli. In the normal state, the protein *LexA* is known to be a repressor to the SOS genes. When DNA is damaged, the protein *RecA* becomes activated and mediates *LexA* autocleavage, which causes activation of the SOS genes. After the activated SOS genes repair the damaged DNA, *RecA* stops mediating *LexA* autocleavage and *LexA* represses the SOS genes again. The full SOS DNA repair gene network is displayed in Figure 6 [45, 28]. We attempt to infer this network from gene expression time series datasets generated by [39] (http://www.weizmann.ac.il/mcb/UriAlon/download/downloadable-data). Each time series contains 50 measurements for every 6 minutes including the initial zero concentrations; we pick the third dataset in the database for this experiment, and compare the results against those found in [31, 30, 28].

The sparsity parameter λ in (10) is chosen to produce about half of the possible edges in the six-gene network. Figure 6 displays in red the edges of the original network that were successfully recovered by a consensus of the top three networks found by PALLAS, according to the penalized likelihood score (the full network is displayed in the Supplementary Material). We can see that all inhibitory edges from *lexA* were successfully detected. Although PALLAS infers the wrong direction between *recA* and *lexA*, the connection is detected. With a similar total number of inferred edges, [30] finds the opposite regulations, i.e., all the inhibitory edges are inferred as activating edges. While [31] finds most of the inhibitory edges, it misses the important edge from *lexA* to *uvrA*. Finally, [28] recovers only two of the edges.

3.3.2 E. Coli Biofilm Formation Pathway

In this section, we demonstrate the performance of PALLAS on RNA-Seq time series expression data from a pathway involved in biofilm formation by E. Coli, namely, the Rpos(sigmaS)/MlrA/CsgD cascade, which involves eight genes: *Rpos, MlrA, CsgD, YciR, YoaD, BcsA, YaiC, YdaM.* Information on this pathway can be found in the KEGG database (https://www.genome.jp/kegg/) as well as in [22, 35, 36]. Figure 7 displays a consensus gene network derived from these sources. The gene expression data used is from the E. Coli Strain B/REL606 and is available at the Dryad Digital Repository (https://datadryad.org/resource/doi:10.5061/dryad.hj6mr) [23]. This dataset consists of 3 bacterial samples and 9 time points evenly spaced for each sample. The genes in this pathway display similar values at low expression levels, but vary considerably at high expression levels. Accordingly, we assume a single baseline parameter $\mu_i \equiv \mu$ for all genes, but the parameters δ_i and ϕ_i are allowed to differ from gene to gene, for $i = 1, \ldots, 8$. The sequencing depth is set at $s = 1.02$ (1k-50k reads) reflecting the low read counts in the data set.

As in the previous experiment, the sparsity parameter λ in (10) is chosen to produce about half of the possible edges in the eight-gene network. Figure 7 displays in red the edges of the original network that were successfully recovered by a consensus of the top three networks found by PALLAS, according to penalized likelihood score (the full network is displayed in the Supplementary Material). PALLAS successfully infers five out of the six important activating interactions from *RpoS*. Most of the other connections in the original network were correctly detected.

4 Conclusion

We presented in this paper PALLAS, a new framework for inference of Boolean gene regulatory networks from gene expression time series data. The algorithm avoids ad-hoc binarization of the gene expression data and allows inference of large networks by employing penalized maximum likelihood as a regularization method, applying particle filtering for the computation of the likelihood, and using a novel version of the fish school search

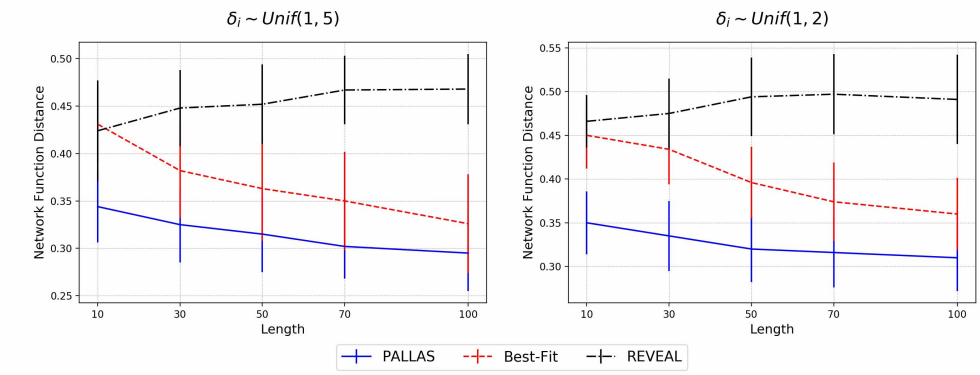

Figure 4: Comparison of network function distance among the PALLAS, Best-Fit, and REVEAL algorithms, under different δ ranges.

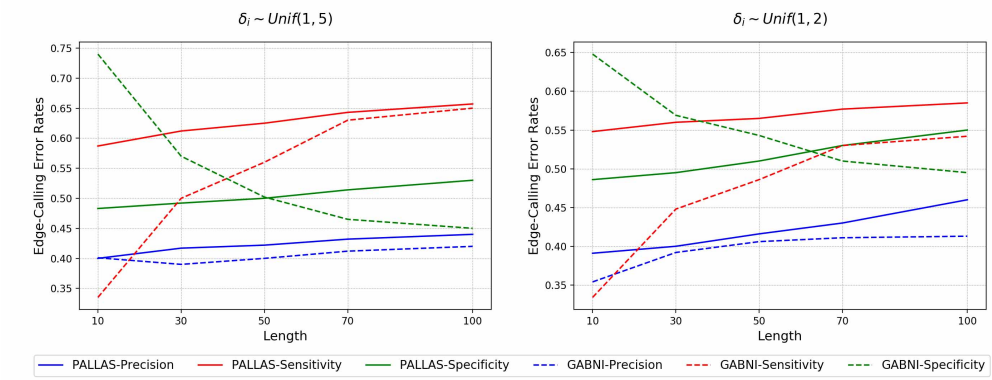

Figure 5: Comparison of edge-calling error rates between the PALLAS and GABNI algorithms, under different δ ranges.

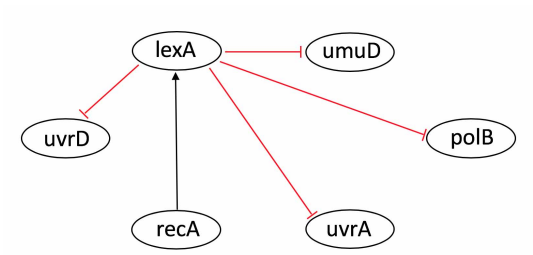

Figure 6: SOS DNA repair system in E.coli (the red edges are the ones successfully recovered by PALLAS).

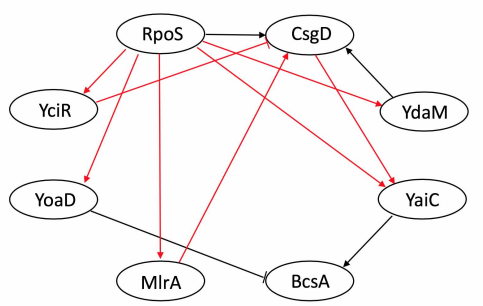

Figure 7: Biofilm architecture of Escherichia coli (the red edges are the ones successfully recovered by PALLAS).

particle swarm algorithm to search the parameter space. Numerical experiments using synthetic time series data show that PALLAS outperforms other well-known GRN inference methods. The performance of PALLAS was also demonstrated on real gene expression time series data from the SOS DNA repair and Biofilm formation pathways in E. Coli.

References

[1] Réka Albert and Hans G Othmer. The topology of the regulatory interactions predicts the expression pattern of the segment polarity genes in drosophila melanogaster. *Journal of theoretical biology*, 223(1):1–18, 2003.

[2] Simon Anders and Wolfgang Huber. Differential expression analysis for sequence count data. *Genome biology*, 11(10):R106, 2010.

[3] M Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on signal processing*, 50(2):174–188, 2002.

[4] Shohag Barman and Yung-Keun Kwon. A novel mutual information-based boolean network inference method from time-series gene expression data. *PloS one*, 12(2):e0171097, 2017.

[5] Shohag Barman and Yung-Keun Kwon. A boolean network inference from time-series gene expression data using a genetic algorithm. *Bioinformatics*, 34(17):i927–i933, 2018.

[6] Carmelo JA Bastos Filho, Fernando B de Lima Neto, Anthony JCC Lins, Antonio IS Nascimento, and Marilia P Lima. A novel search algorithm based on fish school behavior. In *Systems, Man and Cybernetics, 2008. SMC 2008. IEEE International Conference on*, pages 2646–2651. IEEE, 2008.

[7] CJA Bastos-Filho and DO Nascimento. An enhanced fish school search algorithm. In *Computational Intelligence and 11th Brazilian Congress on Computational Intelligence (BRICS-CCI & CBIC), 2013 BRICS Congress on*, pages 152–157. IEEE, 2013.

[8] Natalie Berestovsky and Luay Nakhleh. An evaluation of methods for inferring boolean networks from time-series data. *PloS one*, 8(6):e66031, 2013.

[9] Ulisses Braga-Neto. Optimal state estimation for boolean dynamical systems. In *2011 Conference Record of the Forty Fifth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, pages 1050–1054. IEEE, 2011.

[10] Ting Chen, Hongyu L He, and George M Church. Modeling gene expression with differential equations. In *Biocomputing'99*, pages 29–40. World Scientific, 1999.

[11] Daizhan Cheng and Hongsheng Qi. A linear representation of dynamics of boolean networks. *IEEE Transactions on Automatic Control*, 55(10):2251–2258, 2010.

[12] Riet De Smet and Kathleen Marchal. Advantages and limitations of current network inference methods. *Nature Reviews Microbiology*, 8(10):717, 2010.

[13] Patrik D'haeseleer, Xiling Wen, Stefanie Fuhrman, and Roland Somogyi. Linear modeling of mrna expression levels during cns development and injury. In *Biocomputing'99*, pages 41–52. World Scientific, 1999.

[14] Edward R Dougherty. Validation of inference procedures for gene regulatory networks. *Current genomics*, 8(6):351–359, 2007.

[15] Michael B Eisen, Paul T Spellman, Patrick O Brown, and David Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences*, 95(25):14863–14868, 1998.

[16] A. Faure, A. Naldi, C. Chaouiya, and D. Thieffry. Dynamical analysis of a generic boolean model for the control of the mammalian cell cycle. *"Bionformatics"*, 22(14):e124–e131, 2006.

[17] Adrien Fauré, Aurélien Naldi, Claudine Chaouiya, and Denis Thieffry. Dynamical analysis of a generic boolean model for the control of the mammalian cell cycle. *Bioinformatics*, 22(14):e124–e131, 2006.

[18] Nir Friedman, Michal Linial, Iftach Nachman, and Dana Pe'er. Using bayesian networks to analyze expression data. *Journal of computational biology*, 7(3-4):601–620, 2000.

[19] Noushin Ghaffari, Mohammadmahdi R Yousefi, Charles D Johnson, Ivan Ivanov, and Edward R Dougherty. Modeling the next generation sequencing sample processing pipeline for the purposes of classification. *BMC bioinformatics*, 14(1):307, 2013.

[20] Thomas J Hardcastle and Krystyna A Kelly. baseq: empirical bayesian methods for identifying differential expression in sequence count data. *BMC bioinformatics*, 11(1):422, 2010.

[21] Anne-Claire Haury, Fantine Mordelet, Paola Vera-Licona, and Jean-Philippe Vert. Tigress: trustful inference of gene regulation using stability selection. *BMC systems biology*, 6(1):145, 2012.

[22] Regine Hengge. Principles of c-di-gmp signalling in bacteria. *Nature Reviews Microbiology*, 7(4):263, 2009.

[23] John R Houser, Craig Barnhart, Daniel R Boutz, Sean M Carroll, Aurko Dasgupta, Joshua K Michener, Brittany D Needham, Ophelia Papoulas, Viswanadham Sridhara, Dariya K Sydykova, et al. Controlled measurement and comparative analysis of cellular components in e. coli reveals broad regulatory changes in response to glucose starvation. *PLoS computational biology*, 11(8):e1004400, 2015.

[24] Sui Huang. Gene expression profiling, genetic networks, and cellular states: an integrating concept for tumorigenesis and drug discovery. *Journal of molecular medicine*, 77(6):469–480, 1999.

[25] Mahdi Imani and Ulisses M Braga-Neto. Maximum-likelihood adaptive filter for partially observed boolean dynamical systems. *IEEE Transactions on Signal Processing*, 65(2):359–371, 2017.

[26] Mahdi Imani and Ulisses M Braga-Neto. Particle filters for partially-observed boolean dynamical systems. *Automatica*, 87:238–250, 2018.

[27] Alexandre Irrthum, Louis Wehenkel, Pierre Geurts, et al. Inferring regulatory networks from expression data using tree-based methods. *PloS one*, 5(9):e12776, 2010.

[28] Md Julfikar Islam, Tanveer M.S.R., and Akhand M.A.H. Gene regulatory network inference using prominent swarm intelligence methods. *Computational Biology and Bioinformatics*, 4(5):37–44, 2016.

[29] Stuart A Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of theoretical biology*, 22(3):437–467, 1969.

[30] Shuhei Kimura, Satoshi Nakayama, and Mariko Hatakeyama. Genetic network inference as a series of discrimination tasks. *Bioinformatics*, 25(7):918–925, 2009.

[31] Shuhei Kimura, Katsuki Sonoda, Soichiro Yamane, Hideki Maeda, Koki Matsumura, and Mariko Hatakeyama. Function approximation approach to the inference of reduced ngnet models of genetic networks. *BMC bioinformatics*, 9(1):23, 2008.

[32] Harri Lähdesmäki, Ilya Shmulevich, and Olli Yli-Harja. On learning gene regulatory networks under the boolean network model. *Machine learning*, 52(1-2):147–167, 2003.

[33] Fangting Li, Tao Long, Ying Lu, Qi Ouyang, and Chao Tang. The yeast cell-cycle network is robustly designed. *Proceedings of the National Academy of Sciences*, 101(14):4781–4786, 2004.

[34] Shoudan Liang, Stefanie Fuhrman, and Roland Somogyi. Reveal, a general reverse engineering algorithm for inference of genetic network architectures, 1998.

[35] Franziska Mika and Regine Hengge. Small regulatory rnas in the control of motility and biofilm formation in e. coli and salmonella. *International journal of molecular sciences*, 14(3):4560–4579, 2013.

[36] Franziska Mika and Regine Hengge. Small rnas in the control of rpos, csgd, and biofilm architecture of escherichia coli. *RNA biology*, 11(5):494–507, 2014.

[37] Kevin Murphy, Saira Mian, et al. Modelling gene expression data using dynamic bayesian networks. Technical report, Technical report, Computer Science Division, University of California, Berkeley, CA, 1999.

[38] Hongsheng Qi and Daizhan Cheng. Analysis and control of boolean networks: A semi-tensor product approach. In *2009 7th Asian Control Conference*, pages 1352–1356. IEEE, 2009.

[39] Michal Ronen, Revital Rosenberg, Boris I Shraiman, and Uri Alon. Assigning numbers to the arrows: parameterizing a gene regulation network by using accurate expression kinetics. *Proceedings of the national academy of sciences*, 99(16):10555–10560, 2002.

[40] João AG Sargo, Susana M Vieira, João MC Sousa, and Carmelo JA Bastos Filho. Binary fish school search applied to feature selection: Application to icu readmissions. In *Fuzzy Systems (FUZZ-IEEE), 2014 IEEE International Conference on*, pages 1366–1373. IEEE, 2014.

[41] Ilya Shmulevich, Edward R Dougherty, Seungchan Kim, and Wei Zhang. Probabilistic boolean networks: a rule-based uncertainty model for gene regulatory networks. *Bioinformatics*, 18(2):261–274, 2002.

[42] Ilya Shmulevich, Edward R Dougherty, and Wei Zhang. From boolean to probabilistic boolean networks as models of genetic regulatory networks. *Proceedings of the IEEE*, 90(11):1778–1792, 2002.

[43] Ilya Shmulevich, Ilya Gluhovsky, Ronaldo F Hashimoto, Edward R Dougherty, and Wei Zhang. Steady-state analysis of genetic regulatory networks modelled by probabilistic boolean networks. *Comparative and functional genomics*, 4(6):601–608, 2003.

[44] V Anne Smith, Jing Yu, Tom V Smulders, Alexander J Hartemink, and Erich D Jarvis. Computational inference of neural information flow networks. *PLoS computational biology*, 2(11):e161, 2006.

[45] Mark D Sutton, Bradley T Smith, Veronica G Godoy, and Graham C Walker. The sos response: recent insights into umudc-dependent mutagenesis and dna damage tolerance. *Annual review of genetics*, 34, 2000.

[46] Yukun Tan, Fernando B Lima Neto, and Ulisses Braga Neto. Inference of gene regulatory networks by maximum-likelihood adaptive filtering and discrete fish school search. In *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2018.

[47] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[48] Eugene P van Someren, Lodewyk FA Wessels, and Marcel JT Reinders. Linear modeling of genetic networks from experimental data. In *Ismb*, pages 355–366, 2000.

[49] Daniel C Weaver, Christopher T Workman, and Gary D Stormo. Modeling regulatory networks with weight matrices. In *Biocomputing'99*, pages 112–123. World Scientific, 1999.

[50] Rui Xu, Ganesh K Venayagamoorthy, and Donald C Wunsch II. Modeling of gene regulatory networks with hybrid differential evolution and particle swarm optimization. *Neural Networks*, 20(8):917–927, 2007.