Computing Minimal Boolean Models of Gene Regulatory Networks

Guy Karlebach ,[1*] Peter N Robinson ,[1]

[1]Fitchburg State University, Fitchburg, MA 01420, USA

[*]To whom correspondence should be addressed;

E-mail: gkarleba@fitchburgstate.edu

October 14, 2024

Keywords: Boolean Network, Gene Regulation and Modeling

Abstract: Models of Gene Regulatory Networks (GRNs) capture the dynamics of the regulatory processes that occur within the cell as a means to understanding the variability observed in gene expression between different conditions. Arguably the simplest mathematical construct used for modeling is the Boolean network, which dictates a set of logical rules for transition between states described as Boolean vectors. Due to the complexity of gene regulation and the limitations of experimental technologies, in most cases knowledge about regulatory interactions and Boolean states is partial. In addition, the logical rules themselves are not known a-priori. Our goal in this work is to create an algorithm that finds the network that fits the data optimally, and identify the network states that correspond to the noise-free data. We present a novel methodology for integrating

experimental data and performing a search for the optimal consistent structure via optimization of a linear objective function under a set of linear constraints. In addition, we extend our methodology into a heuristic that alleviates the computational complexity of the problem for datasets that are generated by single-cell RNA-Sequencing (scRNA-Seq). We demonstrate the effectiveness of these tools using simulated data, and in addition a publicly available scRNA-Seq dataset and the GRN that is associated with it. Our methodology will enable researchers to obtain a better understanding of the dynamics of gene regulatory networks and their biological role.

1 Introduction

Maintenance of cellular functions requires the orchestration of many interleaving processes over time and space. A Gene Regulatory Network (GRN) is a set of genes such that the present state of their expression trajectory can be predicted from past states via the regulatory relationships between the genes. As a model that can display complex behavior and at the same time is straightforward to specify, GRNs have been used to describe the regulation of process as different as cell differentiation, circadian clocks and diauxic shift (Geistlinger et al., 2013; Ohara et al., 2018; Willis and Nutt, 2019). Consequently, many methods for reconstructing GRNs from experimental data at varying levels of detail have been proposed (Karlebach and Shamir, 2008; Shavit et al., 2016; Hashimoto et al., 2004). Arguably the most basic formulation, the Boolean network, describes gene expression states as Boolean values and changes in those levels as Boolean functions (Kauffman, 1969). While the simplicity of this model imposes a certain level of abstraction, it also makes it applicable to a broader range of datasets and simplifies its analysis. Interestingly, despite the relative simplicity

of Boolean networks, fitting a Boolean network to a gene expression dataset given a set of regulatory relationships is an NP-Hard problem (Karlebach and Shamir, 2012). Furthermore, while individual regulatory interactions may be available from previous studies, the interactions that occur in a given dataset are not known in advance, which can result in redundant regulators after fitting. Therefore, an algorithm that selects the optimal Boolean network with respect to the input data while taking into account the computational complexity of the problem is needed(Schwarz, 1978; Akaike, 1998; Rissanen, 1983). Laehdesmaeki et al. proposed an algorithm for selecting Boolean functions that fit the data with minimum error (Lähdesmäki, 2003). In this approach functions are selected independently and the goal is to minimize the error of individual functions using the available regulators. Liang et al. suggested an approach that addresses the same objective, and uses mutual information between regulators and target in order to select the regulators of a gene (Liang et al., 1998). Akutsu et al. examined the problem of learning regulation functions when only partial experimental data are available, and showed that it is NP-complete (Akutsu et al., 2009). Barman et al. used a genetic algorithm to search for regulatory functions that preserve a steady-state (Barman and Kwon, 2018). Han et al. formulated the network inference problems as a Bayesian model inference problem and suggested a Markov Chain Monte Carlo algorithm for finding the posterior given data (Han et al., 2014). Karlebach and Shamir proposed a method to find a model that fits the observed noisy Boolean trajectories that minimizes the conditional entropy of the gene targets given their regulators (Karlebach and Shamir, 2012). In this paper, we combine the problems of edge selection, logic optimization and data de-noising in a single objective function. We further present a novel algorithm for optimizing this objective, i.e. finding the optimal Boolean network with respect to its fit to a dataset and de-noising the data. We

show that the optimal structure conforms with a minimal encoding and noise representation which we refer to as minimal edit distance from a state of ignorance. Our approach addresses the computational complexity of the problem by formulating it as 0/1 Integer Linear Programming. Thus, it can be solved using powerful branch and bound methods that were developed for ILP. In addition, we provide a heuristic that can be used to solve the problem or as a subroutine for finding bounds for the 0/1 ILP solution. We demonstrate the effectiveness of our methodology on a gene expression dataset of single-cell RNA Sequencing from mouse embryonic midbrain.

2 Methods

Optimal Fit of Boolean Networks

A gene expression dataset consists of a N $\times$ M matrix, where N corresponds to the number of genes whose expression level was measured, and M corresponds to the number of experiments. The expression values in the matrix can be discrete or continuous, depending on the experimental technology that was used for generating the data, and in both cases captures the variation in transcriptional regulation and contains noise. In order to map these values to Boolean values one needs to label each observation as belonging to a state of low or high expression. The mapping does not need to be perfect, since any analysis method should be able to account for some degree of incorrect mappings as a result of noise. As the methodology proposed in this section is independent of the choice of a mapping, in the rest of this section we will assume that the mapping has already been applied to the data. In the Results section we demonstrate the process using an experimental dataset.

A trajectory of a Boolean network is a sequence of states such that each

state except for the first state in the sequence is derived from the previous state using a set of Boolean functions, also known as logic tables. Each Boolean function determines the value of exactly one gene, and its inputs are the Boolean values of any number of genes in the network. Usually, it is assumed that the number of inputs is small compared to the total number of genes. The regulatory relationships of a Boolean network can be illustrated as edges from inputs to outputs in a directed graph, called a regulation graph. A gene that has an outgoing edge to another gene is referred to as a regulator, and a gene with an incoming edge as a target (a gene can be both a regulator and a target in the same network). A steady state is a state that repeats itself in a trajectory indefinitely unless perturbed by external signals, i.e. signals that are not part of the network. In a typical gene expression dataset the experiments correspond to a single time point, and therefore the network is assumed to be in a steady state in each experiment. For simplicity of description we assume in the rest of this section that the network is in steady state, however the algorithm presented here is applicable to time-series as well. This is easy to see if we convert the trajectory to an undirected regulation graph, where each data point corresponds to a node/gene, and a node/gene in time t is regulated by the nodes at time t-1 that correspond to its regulators in the original regulation graph. Then the same variables describe the logic tables and regulators of all the copies of the same gene, i.e. the nodes that correspond to it in different time points. We demonstrate the use of time-series data in the results section.

Discrepancies between a dataset and a network model occur when the Boolean values in an experimental dataset do not agree with any network trajectory due to experimental noise. This presents a difficulty if the network model is not known a-priori, since enumerating all possible networks is infeasible. Formally, let g_i denote the integer index of the i^{th} gene in the list of genes, let $C_{g_i,j}$ denote

the Boolean value of gene g_i in experiment j, and let e_{g_1,g_2} denote a directed edge between genes g_1 and g_2. We say that the data contains a discrepancy if for some gene index g and two experiments i_1 and i_2, $C_{g_j,i_1} = C_{g_j,i_2}$ for all genes g_j such that an edge $e_{g_j,g}$ exists, but $C_{g,i_1} \neq C_{g,i_2}$. It follows from the network's determinism that at least one of the experiments i_1 or i_2 does not agree with any network trajectory. To simplify the notation, we will refer to a gene by its name or index in the list of genes interchangeably. The number of regulators of a gene g will be denoted as $indegree(g)$.

Assuming that $P \neq NP$, there does not exist a polynomial time algorithm for resolving all the discrepancies with the minimal number of changes. Therefore, either a heuristic that finds a local optimum or an algorithm that may not terminate in a reasonable amount of time must be used instead. Another difficulty is that a strict subset of the regulation graph may provide a better fit to the data, as not all the interactions occur under every condition, and so the structure of the network itself needs to be considered in the search for the optimal solution. If every binary string encodes a network, then every additional bit used for network encoding doubles the number of possible networks. Similarly, every bit of input that we allow a mismatch in doubles the number of trajectories the networks can match. In order to choose a solution that is a-priori as likely to add a network bit as it is to edit a bit of input, we would like to assign the same cost to every network/edit bit. As it turns out, the problem can be formulated as 0/1 ILP, as described in the next subsection.

An Algorithm for the Optimal Minimal Network

The in-degree of nodes in the regulation graph is usually assumed to be small compared to the number of genes or the number of experiments. If we assume that it is a constant value in terms of computational complexity, we can define

a set of constraints on the values that have to be changed in order to remove all discrepancies from the data. Whenever input values are observed to equal a line in the logic table whose output is 1, or otherwise mapped to such a line by correcting noise, the observed output, or corrected output, should also be 1. If the observed output is not 1, then either the observed value that matched the line were affected by noise, or their mapping to that line should be changed. The case of logic table lines whose output was determined to be 0 can be handled similarly. We define these constraints as follows: . Let $C_{i,j}$ denote the Boolean input value of gene i at experiment j, and let $B_{g_i,j}$ equal 1 if the value of gene g_i in experiment j was flipped in the solution (i.e. did not fit the trajectories of the inferred network), and otherwise 0. Then for every experiment j and for every gene g_{k+1} with regulators $g_1, g_2, ..., g_k$ and for every Boolean vector $(w_1, w_2, ..., w_k)$, $w_j \in \{0, 1\}$, if the output of the Boolean function that determines the value of g_{k+1}, $I(w_1, w_2, ..., w_k)$, is 1, the following constraint must hold:

$$\sum_{r=1}^{k}(C_{r,j} \cdot (w_r + (1 - 2 \cdot w_r) \cdot B_{g_r,j}) \tag{1}$$

$$+ (1 - C_{r,j}) \cdot ((1 - w_r) + (2 \cdot w_r - 1) \cdot B_{g_r,j}))$$

$$+ C_{k+1,j} \cdot B_{g_{k+1},j} + (1 - C_{k+1,j}) \cdot (1 - B_{g_{k+1},j})$$

$$< (2 - I(w_1, w_2, ..., w_k)) \cdot (k + 1)$$

This constraint means that if the output variable $I(w_1, w_2, ..., w_k)$ was set to 1, whenever the inputs $w_1, w_2, ..., w_k$ appear in the solution, the output (the value of g_{k+1}) must be 1. The values $w_1, w_2, ..., w_k$ are the values of the regulators in the line of the logic table that is considered in the constraint. They would have as many values as the logic table can have rows, in other words $2^{indegree(g_{k+1})}$,

and for each such assignment of values we will create one constraint. Similarly, if $I(w_1, w_2, ..., w_k)$ is set to 0 the following constraint must hold:

$$\sum_{r=1}^{k}(C_{r,j} \cdot (w_r + (1 - 2 \cdot w_r) \cdot B_{g_r,j}) \tag{2}$$

$$+ (1 - C_{rj}) \cdot ((1 - w_r) + (2 \cdot w_r - 1) \cdot B_{g_r,j}))$$

$$+ C_{k+1,j} \cdot (1 - B_{g_{k+1},j}) + (1 - C_{k+1,j}) \cdot B_{g_{k+1},j}$$

$$< (I(w_1, w_2, ..., w_k) + 1) \cdot (k + 1)$$

By requiring that under these constraints the following sum is minimized:

$$\sum {}^{i\in 1,..,N}_{j\in 1,..,M} B_{ij}$$

we can use a branch and bound algorithm for 0/1 integer programming to find a solution that fits the data with a minimal number of changes, and construct a new dataset with values $D_{ij} = ((C_{ij} + B_{ij}) \mod 2), i \in 1..N, j \in 1..M$.

However, this formulation assumes that all regulatory interactions take effect in the data, which is rarely the case in practice. In order to choose the solution that also minimizes the network structure, for every gene g_i and each one of its targets g_j, we create another Boolean variable R_{ij}, that is constrained to be greater equal than every difference between regulatory outputs (the I variables) where pairs of logic table rows are identical in all inputs except g_i. If the regulatory output changes when only the regulator g_i changes its value, it will be constrained to equal 1. Since other inputs can be removed as well, we add the same constraints to R_{ij} even for pairs of rows where other inputs change,

and we add the R variables of the changing inputs with a minus sign to the right hand side, i.e subtract them from the difference between the I variables. If the other variables that change have been removed, R_{ij} will still be subjected to the same constraint, and will be set to 1 if the output changes. If the new constraints can be satisfied without setting R_{ij} to 1, then the edge from g_i to g_j in the regulation graph is redundant, because it does not explain any change that is not explained by other edges. Intuitively, whenever a change in output occurs between two lines of the logic table, at least one regulator that changes values between those lines should explain the difference. If there is a change, then one line's output is 1 and the other is 0, and so one of the differences between the outputs will be equal 1. If we rearrange the terms in the constraint such that all R variables are on the left hand side and all I variables on the right hand side, we see that the sum of R variable is greater than the difference between the first I variable and the second I variable in one constraint, and the sum of R variables is greater than the difference between the second I variable and the first I variable in another constraint. This equivalent formulation matches the intuitive notion of explaining differences between logic table lines. Hence in total, one constraint is added for each pair of rows of a logic table. Using the R_{ij} variables, we can also account for the size of the logic tables they contribute to, by defining a new type of variable $V_{R_k} : k \in 1..indegree(g)$ that sums the number of R_{ij} variables of a gene g (the gene index is removed from the variable name for simplicity). This variable will be constrained to 1 if at least R of the variables if gene g are set to 1. For example, for a gene with 3 regulators, V_{R_3} is a Boolean variable which is greater or equal the mean of the three regulator variables minus 2/indegree(g), so it is only constrained to equal 1 if all three regulators of the gene are kept. Similarly, V_{R_2} is a Boolean variable which is greater or equal than the mean of the regulator variables of

the gene minus $1/indegree(g)$, and V_{R_1} is a Boolean variable greater or equal than the mean of the regulator variables minus $0.5/indegree(g)$. The weights of these variables in the objective function are 1 for V_{R_1}, $2^2 - 1$ for V_{R_2} and $2^3 - 2^2$ for V_{R_3}. If V_{R_3} is equal 1, so do the other two variables, and therefore the objective function is added with the size of a Boolean table of 3 regulators, i.e 2^3. Similarly, if V_{R_3} is 0 but V_{R_2} is equal 1, so is V_{r_1}, and so the addition to the objective function is 2^2. If a single regulator is chosen, only one bit is needed to represent the logic table, and that is the addition to the objective function. This example is trivially generalized to any number of regulators greater than 1, by requiring that the i^{th} V variable be greater or equal to the difference between the mean of the R variables and $(i-1)/indegree(g)$, and setting the weight to the difference between the corresponding size of the truth table and that size for i-1 regulators. The number of variables in the resulting 0/1 integer linear programming is $M \cdot N + \sum_{i=1}^{N} [2^{indegree(g_i)}] + 2 * |E|$. We observe that the first regulator added to a gene has N possible choices, and thus is equivalent to adding $log_2(N)$ bits to the network representation. The second regulator adds $log_2((N-1)/2)$, bits and so on with $log_2((N-i+1)/i)$ bits for the i^{th} regulator. The sum of these weights can be added through addition to the respective V variable weights. A truth table with one regulator adds a factor of two to the number of networks (activation or repression), and therefore adds 1 bit to the representation. A truth table with two regulators add approximately 4 bits (since some tables correspond to less than two regulators), a truth table with 3 regulators approximately 8 bits and so on, and therefore we add $\sim 2^i$ to the weight of a gene's V_i variable, and apply telescopic cancellation by subtracting the weight of the previous V variable, as discussed earlier. Finally, since each bit of noise reduces the number of noise-free inputs by a factor of 2, we set the weights of the B variables to 1. All other variables have weight 0. Minimization

of the objective will then minimize the edit distance from a state of an empty network and noise, as desired.

Every optimal solution remains optimal for the same input with reduced noise. Suppose that network of $k > 0$ bits is the optimal solution with r mismatches. Now we flip one of the mismatches, and assume towards contradiction that there is another network that becomes the optimal solution. This means that the new network matches a trajectory that is identical to the previous one except for one place, and the sum of its encoding bits and mismatches is strictly smaller than k+r-1. Therefore, the same network would have fit the original trajectory, which only differs in one bit, by at most k+r-1, which contradicts the assumption that the optimal solution fits with cost k+r. This argument can be applied repeatedly to generate 2^r trajectories that an optimal solution must fit. Consequently, there cannot be more than 2^{L-r} solutions that use k network bits and r noise bits, where L is the length of the input. So if $k << L - r$, the fit to the data would not be random(Kolmogorov, 1968).

A Heuristic for Single Cell Datasets

Speeding up the solution for computationally hard problems has been an active topic of Bioinformatics research of scRNA-Seq (Seth et al., 2022; Liu et al., 2017). A heuristic for solving the fitting problem would enable fitting for hard instances, and could also be used for improving the upper bound during the 0/1 ILP search. Therefore, we propose the following heuristic for scRNA-Seq data:

1. Cluster the input states into K clusters.

2. Solve the problem for the set of cluster centers, where fractional values in the centers are rounded to the closer bit, to obtain the set of regulators E, and find the de-noised states using the fixed set E as described in the previous section, to obtain a de-noised set of states H^*.

3. Initialize the full solution H to the empty set. For input state $i \in 1..M$, add it to H^* and if it conflicts with a state in H^* with respect to E, change it incrementally to match the entries of the state in H^* that is closest to it, until all discrepancies are resolved, and then add it to the full solution H as well. Otherwise, if it does not conflict add it to H and H^* without change.
4. Return H

By the construction of H^*, it is free from discrepancies. H is built incrementally such that after every addition of a state it is free from discrepancies, and therefore it is also free of discrepancies. It is easy to see that when step 3 is completed, either a new state that is free of discrepancies or an existing state, that by the loop invariant is free of discrepancies, is added to H and H^*. Similarly, step 3 preserves the set of regulators inferred in step 2, and therefore it does not increase the number of regulators. In single-cell data, clusters of cells will have a similar network state, and thus it is likely that a close state to the one that is present in the optimal solution will already be included in H^*. This will provide an upper bound for the number of changes applied to each state. Note that the optimal order by which changes are applied to the conflicting state in step 3 may depend on the input. For example, one may wish to order the changes according to the number of discrepancies that they resolve after the last change, or to choose the order based on the network structure. Similar considerations can be applied to the order by which states are selected for addition, for example, by the number of discrepancies with states that have not been added yet. The weight of a discrepancy in step 1 can be set to the number of states in its cluster, as each cluster represents a set of states that are fitted to the network's trajectories. Then, step 2 can be performed using an ILP solver and the formulation described in the previous section. In principle, this

step can also be performed recursively until a set of consistent states, with respect to the input set of regulators, is obtained. The clustering in each recursive call will decrease the number of states, and so in the worst case the recursion will halt when we reach a single state, because it does not contain discrepancies. If the heuristic is using trajectories as input, a single trajectory can be rid of discrepancies by flipping the bits that generate them from the latest time points backwards. Regulators whose removal would cause discrepancies after step 4 are then kept, and the rest removed, for example by backward selection. Note. however, that the recursive approach with an arbitrary implementation can result in a poor approximation of the optimal solution.

Results

Simulation

The algorithm described in the previous section finds the structure and denoising that correspond to the minimal edit distance from a state of ignorance criterion, and addresses the computational complexity of this problem. The following section examines the quality of the optimal solution compared to solutions that are not optimal with respect to this criterion. To this end, we used the function 'generateRandomNKnetworks' in the R package BoolNet (Müssel et al., 2010) to generate random Boolean networks and their noisy trajectories. BoolNet also provides implementations of two network inference methods , BESTFIT (Lähdesmäki, 2003) and REVEAL (Liang et al., 1998) , to compare the optimal solution to. generated 100 random Boolean networks and time series data using the BoolNet R package. Each network had 5 genes, 2 regulators per gene, 10 time series and 20 time points per series. In addition, for each gene we randomly added an edge that does not belong to the true structure, and added

Bernoulli noise to the time series with p=0.1. Gurobi (Gurobi Optimization, LLC, 2023) was used to solve using our method and the BESTFIT method in the BoolNet package was used as comparison. The implementation of the REVEAL method in the same package did not support noisy data. Figure 1 shows boxplots of the number of true positives and the number of false positives for both tools. As can be seen in the figure, our method (denoted MEDSI for Minimum Edit Distance from a State of Ignorance) has a higher rate of true positives and lower rate of false positives. In fact, it seldom chooses wrong edges. We repeated this analysis with different parameters for the random datasets - increasing the noise to 0.15 and decreasing the dataset size to 5 time-series of 10 time points each, the median number of true positives predicted by BESTFIT is 7, and the median number of false positives is 4. Since a third of the edges that can be selected are false positives, this performance is not better than randomly choosing edges. In contrast, the median number of true positives of our algorithm was 4.5, and the median number of false positives was 0. This test shows that when the dataset is smaller and noisier, our algorithm is still able to provide reliable predictions. When using 10 genes, 5 series of 10 time points each, and the parameter topology set to 'scale_free', both method predicted a median of 11 true edges, but BESTFIT had a twice as large number of false positives (median of 2 edges vs. 1 edge for our algorithm). We conclude that our algorithm maintains its advantage for different edge distributions as well.

Analysis of the Gene Regulatory Network for Midbrain Dopaminergic Neurons

In order to test our algorithm we use the mouse embryo scRNA dataset of LaManno et al. (Manno et al., 2016) and the midbrain dopaminergic (mDA) neuron developmental GRN that was described by (Arenas et al., 2015). To

obtain the gene counts we used the scRNA R package (Davide Risso, Michael Cole, 2017). The R package Seurat (Satija et al., 2015) provides functions to inspect the data and determine the threshold for screening out cells with an unusually high or low number of features. Specifically, the parameter 'min.cells' of the function 'CreateSeuratObject' was set to 3, removing features that were detected in less than 3 cells. The parameter 'min.features' of this function was set to 200, excluding cells in which less than 200 features were detected. Using the function 'VlnPlot', we next identified outlier cells with respect to a high (4,000 or more) or low (500 or less) number of features, and removed them as well. Using the 'saver' function from the R package SAVER (Huang et al., 2018) with default parameters, we then imputed the network genes. saver has a built-in default normalization, however normalization using the Seurat function 'NormalizeData' with default settings resulted in the same Boolean matrix. After filtering, the dataset contained 1,631 cells (experiments). To obtain Boolean values, for each gene we compute the median expression value, and map values smaller or equal to the median expression value to Boolean 0, and all other counts to 1. The number of clusters K used in the heuristic was set to 50. The clustering algorithm that we used was k-means as implemented in R, with all other parameters at their default values. For solving the 0/1 ILP problem we used Gurobi (Gurobi Optimization, LLC, 2023). We then generated an EBNF description of the network and used the R package BoolNet to analyze the network properties (Müssel et al., 2010). A network whose purpose is to induce multiple different cell states would normally have a higher number of steady states than would be expected from a random network with the same topological properties. In order to test if this is the case for the inferred network, we generated 1,000 random networks with the same number of genes and a scale-free distribution of inputs of a regulation function, using

the 'generateRandomNKNetwork' function with the parameter 'topology' set to 'scale_free' and 'gamma' set to 0.5. Even when considering networks that had at least one steady state, out of 1,000 networks, none had an equal or greater number of steady states than the inferred network, corresponding to a p-value smaller than 0.001. We interpret this result as indicating that the inferred network structure is adapted to supporting a large number of differentiation states. Out of 32 steady states, 28 appeared in the dataset of LaManno et al., suggesting the existence of unobserved phenotypes that may be triggered under similar conditions, or possibly networks states that lead to apoptosis and are therefore not observed in the experiment. The similarity between the observed and unobserved steady states is illustrated in figure 2. States are mapped into a two-dimensional space using multidimensional scaling. There are four clusters of network states, all of them containing observed states (light blue) and three of them containing unobserved (dark blue) states. Each axis corresponds to one coordinate in the 2-dimensional space to which the states are mapped. Multidimensional scaling was performed using the R function 'cmdscale', with the 'dist' function for creating the distance matrix between states. Both functions were called with their default parameters. Examining the activation of genes in the unobserved steady states showed that they mostly belong to several subnetworks based on the division of Arenas et al. (Arenas et al., 2015). These states are shown in figure 3, where an active gene (Boolean 1) is colored red and an inactive gene (Boolean 0) is colored blue. Each row corresponds to a different steady state, and each column corresponds to one gene that has regulators or targets. This could potentially indicate that these states correspond to yet unexplored differentiation pathways that can be triggered by external stimuli without modifying the network components. Next we examined whether the knockout or over-expression of a single gene can generate a new repertoire of

steady states that is not observed in the wild type network. Since knocking down a gene sets its activation value to inactive (Boolean 0), we counted the number of steady states in the perturbed network that differ from each wild type steady state by at least one other gene value. For single-gene knockouts, Ferd3l and Shh generated 16 new steady states, Hes1 and Msx1 generated 12 new steady states, Neurog2 generated 6 new steady states, and Lmx1b generated 4 new steady states. Knockout of any other gene did not generate steady states that differ from wild type steady states by genes that were not knocked down. A similar experiment with gene over-expression resulted in a similar behavior, except that Msx1 generated only 4 new steady states. Figure 4 illustrates these results. The y-axis provides the number of new steady states that result from knocking out (red) or over-expressing(cyan) the genes that are given in the x-axis. These findings suggests that while the network is generally robust to perturbations, there is a subset of genes are potential targets for generating behavior that differs from that of the wild type network. Our in-silico experiments could be repeated in the wet lab in order to further elucidate the connection between network steady states and phenotype. The genes whose perturbations generate the largest number of new states, Shh and Ferd3l, have been identified as potential targets for inducing remyelination and neurogenesis, respectively (Sanchez et al., 2018; Ono et al., 2010). This may indicate that the diversity of the new steady states may reflect an ability to divert the differentiation path into new cell types upon perturbation. Next, we reasoned that cells that were collected at later time points will on average undergo less differentiation than cells collected at earlier time points, since they are more likely to have reached their terminal differentiation states. Therefore, the network that regulates differentiation should have a smaller number of genes that are active. In order to test this hypothesis, we used the R function 'cor.test' to test for correlation

between time point and the number of network genes that are active, using Kendall's test statistic. The p-value was highly significant ($8.3 \cdot 10^{-38}$), with a negative Kendall correlation of -0.24, suggesting that network activity indeed decreases with time. We further tested whether the network's steady states are more divergent than would be expected by chance. Since the cell differentiate into several distinct phenotypes, a larger distance between the regulatory network states is expected. We compared the mean pairwise binary distance between pairs of steady states of the inferred network to the mean distance between pairs of steady states of scale free random networks that had at least five steady states. Out of 1,000 random networks, only 29 had steady states with a larger or equal mean distance, corresponding to a p-value of 0.029.

Conclusion

We propose a new algorithm for fitting a Boolean network model to gene expression data that finds an optimal solution with respect to network structure and fit to the data. We further present a heuristic that alleviates the computational complexity of the problem and therefore provides a practical solution for cases in which an exact solution cannot be obtained due to limited computational resources. Using known regulatory relationships and a dataset of scRNA-Seq measurements, we demonstrated the usefulness of our algorithm by inferring the network structure and its state in different cells. Inspection of the dynamic properties of the inferred network show that only a subnetwork is responsible for generating the observed steady states, and that the dataset only represents a subset of the possible steady states under the experimental conditions. By examining the de-noised data we found that distinct regulatory trajectories could potentially give rise to different types of cells. Single gene perturbations change the steady state behavior significantly only when the targets are a small subset

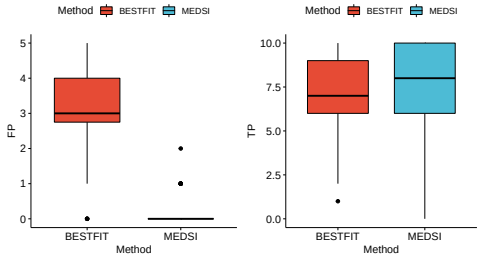

Figure 1: Boxplots of True Positives and False Positives

of the network's genes, suggesting targets for wet lab experiment to explore novel phenotypes. The method presented in this paper provides a novel approach to using regulatory relationships between transcription factors and their targets for the interpretation of gene expression assays and for exploring unobserved regulatory trajectories in-silico. Since the problem described in this work belongs to the class of NP-Complete problems, some instances require significantly more computational resources to solve than problems that have polynomial-time algorithms. The Gurobi solver implements parallelization which can be used to trade running time for computing cores. Additionally, we provided a class of heuristics whose integration with the ILP search to improve running time is one of our future research goals. As discussed in the Methods section, various choices in the implementation of the heuristic can be made and the best choices given a dataset need to be further explored. For example, the optimal parameters choices for the heuristic could be different in different steps of the search. While the Gurobi solver is constantly being improved, it is not currently optimized for the network inference problem. Finally, our method currently collects regulator-target interactions from the literature as a first step. Using a heuristic to provide an equivalent starting point based on expression data alone is another future goal.

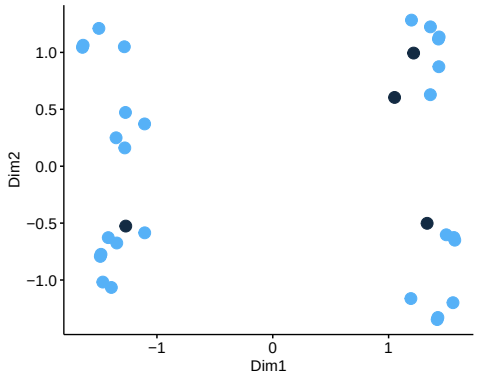

Figure 2: Multidimensional Scaling of Observed and Unobserved Network States

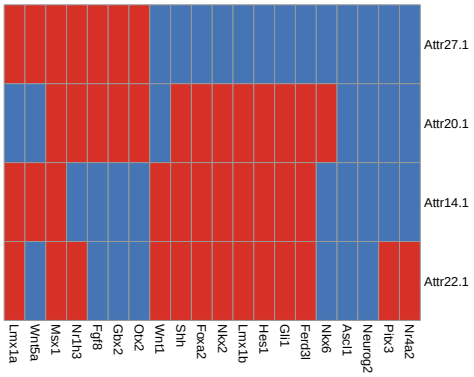

Figure 3: Clustering of Inferred Network States

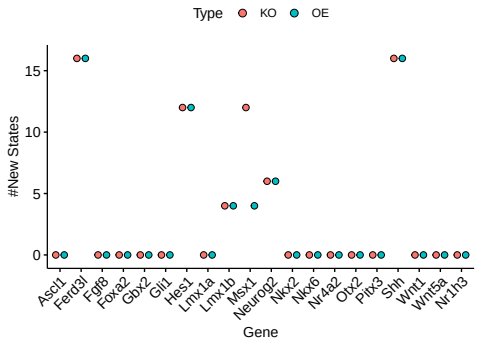

Figure 4: The Number of New Network States after Single Gene Overexpression and Knockout

References

Akaike, H. Information theory and an extension of the maximum likelihood principle. In *Springer Series in Statistics*, pages 199–213. Springer New York, 1998. doi: 10.1007/978-1-4612-1694-0_15. URL https://doi.org/10.1007/978-1-4612-1694-0_15.

Akutsu, T., Tamura, T., and Horimoto, K. Completing networks using observed data. In *Lecture Notes in Computer Science*, pages 126–140. Springer Berlin Heidelberg, 2009. doi: 10.1007/978-3-642-04414-4_14. URL https://doi.org/10.1007/978-3-642-04414-4_14.

Arenas, E., Denham, M., and Villaescusa, J. C. How to make a midbrain dopaminergic neuron. *Development*, 142(11):1918–1936, June 2015. doi: 10.1242/dev.097394. URL https://doi.org/10.1242/dev.097394.

Barman, S. and Kwon, Y.-K. A boolean network inference from time-series gene expression data using a genetic algorithm. *Bioinformatics*, 34(17):i927–i933, September 2018. doi: 10.1093/bioinformatics/bty584. URL https://doi.org/10.1093/bioinformatics/bty584.

Davide Risso, Michael Cole. scrnaseq, 2017. URL https://bioconductor.org/packages/scRNAseq.

Geistlinger, L., Csaba, G., Dirmeier, S., et al. A comprehensive gene regulatory network for the diauxic shift in saccharomyces cerevisiae. *Nucleic Acids Research*, 41(18):8452–8463, July 2013. doi: 10.1093/nar/gkt631. URL https://doi.org/10.1093/nar/gkt631.

Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023. URL https://www.gurobi.com.

Han, S., Wong, R. K. W., Lee, T. C. M., et al. A full bayesian approach for boolean genetic network inference. *PLoS ONE*, 9(12):e115806, December 2014. doi: 10.1371/journal.pone.0115806. URL https://doi.org/10.1371/journal.pone.0115806.

Hashimoto, R. F., Kim, S., Shmulevich, I., et al. Growing genetic regulatory networks from seed genes. *Bioinformatics*, 20(8):1241–1247, February 2004. doi: 10.1093/bioinformatics/bth074. URL https://doi.org/10.1093/bioinformatics/bth074.

Huang, M., Wang, J., Torre, E., et al. SAVER: gene expression recovery for single-cell RNA sequencing. *Nature Methods*, 15(7):539–542, June 2018. doi: 10.1038/s41592-018-0033-z. URL https://doi.org/10.1038/s41592-018-0033-z.

Karlebach, G. and Shamir, R. Modelling and analysis of gene regulatory networks. *Nature Reviews Molecular Cell Biology*, 9(10):770–780, September 2008. doi: 10.1038/nrm2503. URL https://doi.org/10.1038/nrm2503.

Karlebach, G. and Shamir, R. Constructing logical models of gene regulatory networks by integrating transcription factor–DNA interactions with expression data: An entropy-based approach. *Journal of Computational Biology*, 19(1):30–41, January 2012. doi: 10.1089/cmb.2011.0100. URL https://doi.org/10.1089/cmb.2011.0100.

Kauffman, S. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, 22(3):437–467, March 1969. doi: 10.1016/0022-5193(69)90015-0. URL https://doi.org/10.1016/0022-5193(69)90015-0.

Kolmogorov, A. N. Logical basis for information theory and probability theory. *IEEE Trans. Inf. Theory*, IT-14:662–664, 1968.

Lähdesmäki, H. *Machine Learning*, 52(1/2):147–167, 2003. doi: 10.1023/a:1023905711304. URL https://doi.org/10.1023/a:1023905711304.

Liang, S., Fuhrman, S., and Somogyi, R. Reveal, a general reverse engineering algorithm for inference of genetic network architectures. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, pages 18–29, 1998. ISSN 2335-6936. Copyright: This record is sourced from MEDLINE®/PubMed®, a database of the U.S. National Library of Medicine.

Liu, Z., Lou, H., Xie, K., et al. Reconstructing cell cycle pseudo time-series via single-cell transcriptome data. *Nature Communications*, 8(1), June 2017. doi: 10.1038/s41467-017-00039-z. URL https://doi.org/10.1038/s41467-017-00039-z.

Manno, G. L., Gyllborg, D., Codeluppi, S., et al. Molecular diversity of midbrain development in mouse, human, and stem cells. *Cell*, 167(2): 566–580.e19, October 2016. doi: 10.1016/j.cell.2016.09.027. URL https://doi.org/10.1016/j.cell.2016.09.027.

Müssel, C., Hopfensitz, M., and Kestler, H. A. Boolnet – an r package for generation, reconstruction and analysis of boolean networks. *Bioinformatics*, 26(10):1378–1380, 2010.

Ohara, T., Hearn, T. J., Webb, A. A., et al. Gene regulatory network models in response to sugars in the plant circadian system. *Journal of Theoretical Biology*, 457:137–151, November 2018. doi: 10.1016/j.jtbi.2018.08.020. URL https://doi.org/10.1016/j.jtbi.2018.08.020.

Ono, Y., Nakatani, T., Minaki, Y., et al. The basic helix-loop-helix transcription factor nato3 controls neurogenic activity in mesencephalic floor plate cells. *Development*, 137(11):1897–1906, June 2010. doi: 10.1242/dev.042572. URL https://doi.org/10.1242/dev.042572.

Rissanen, J. A universal prior for integers and estimation by minimum description length. *The Annals of Statistics*, 11(2), June 1983. doi: 10.1214/aos/1176346150. URL https://doi.org/10.1214/aos/1176346150.

Sanchez, M. A., Sullivan, G. M., and Armstrong, R. C. Genetic detection of sonic hedgehog (shh) expression and cellular response in the progression of acute through chronic demyelination and remyelination. *Neurobiology of Disease*, 115:145–156, July 2018. doi: 10.1016/j.nbd.2018.04.003. URL https://doi.org/10.1016/j.nbd.2018.04.003.

Satija, R., Farrell, J. A., Gennert, D., et al. Spatial reconstruction of single-cell gene expression data. *Nature Biotechnology*, 33(5):495–502, April 2015. doi: 10.1038/nbt.3192. URL https://doi.org/10.1038/nbt.3192.

Schwarz, G. Estimating the dimension of a model. *The Annals of Statistics*, 6 (2), March 1978. doi: 10.1214/aos/1176344136. URL https://doi.org/10.1214/aos/1176344136.

Seth, S., Mallik, S., Bhadra, T., et al. Dimensionality reduction and louvain agglomerative hierarchical clustering for cluster-specified frequent biomarker discovery in single-cell sequencing data. *Frontiers in Genetics*, 13, February 2022. doi: 10.3389/fgene.2022.828479. URL https://doi.org/10.3389/fgene.2022.828479.

Sharan, R. and Karp, R. M. Reconstructing boolean models of signaling. *Journal of Computational Biology*, 20(3):249–257, March 2013. doi: 10.1089/cmb.2012.0241. URL https://doi.org/10.1089/cmb.2012.0241.

Shavit, Y., Yordanov, B., Dunn, S.-J., et al. Automated synthesis and analysis of switching gene regulatory networks. *Biosystems*, 146:26–34, August 2016. doi: 10.1016/j.biosystems.2016.03.012. URL https://doi.org/10.1016/j.biosystems.2016.03.012.

Willis, S. N. and Nutt, S. L. New players in the gene regulatory network controlling late b cell differentiation. *Current Opinion in Immunology*, 58:68–74, June 2019. doi: 10.1016/j.coi.2019.04.007. URL `https://doi.org/10.1016/j.coi.2019.04.007`.