Computing Minimal Boolean Models of Gene Regulatory Networks

Guy Karlebach
The Jackson Laboratory for Genomic Medicine
Farmington, Connecticut 06032, USA
Email: guy.karlebach@jax.org

Peter N Robinson
The Jackson Laboratory for Genomic Medicine
Farmington, Connecticut 06032, USA
Email: peter.robinson@jax.org

***Abstract*—Models of Gene Regulatory Networks (GRNs) capture the dynamics of the regulatory processes that occur within the cell as a means to understand 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 [3], [11], [18]. Consequently, many methods for reconstructing GRNs from experimental data at varying levels of detail have been proposed [5], [7], [17]. Arguably the most basic formulation, the Boolean network, describes gene expression states as Boolean values and changes in those levels as Boolean functions [9]. 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 [8]. In practice the regulatory interactions that occurs in a given dataset are not known, which can result in redundant regulators after fitting. Not surprisingly, an algorithm for optimal fitting that takes into account the complexity of network structure has not been proposed to-date [1], [12], [15]. In this paper we present a novel algorithm for finding the optimal Boolean network structure with respect to its fit to a dataset, and for de-noising the data. We show that minimizing the encoding of the noise and the network structure can be formulated as a 0/1 Integer Linear Programming problem (ILP), and thus solved using powerful branch and bound methods that exist for ILP. In addition, we provide heuristic that alleviates the computational complexity of the problem, and 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 unedited bit in the input data doubles the number of random strings that correspond to these bits. 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 approximately 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. 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:

$$\sum_{r=1}^{k}(C_{rj} \cdot (w_r + (1 - 2 \cdot w_r) \cdot B_{g_r,j}) \tag{1}$$
$$+(1 - C_{rj}) \cdot ((1 - w_r) + (2 \cdot w_r - 1) \cdot B_{g_r,j}))$$
$$+C_{k+1,j} \cdot B_{g_{k+1},j} + (1 - C_{k+1,j}) \cdot (1 - B_{g_{k+1},j})$$
$$< (2 - I(w_1, w_2, ..., w_k)) \cdot (k + 1)$$

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. Similarly, if $I(w_1, w_2, ..., w_k)$ is set to 0 the following constraint must hold:

$$\sum_{r=1}^{k}(C_{rj} \cdot (w_r + (1 - 2 \cdot w_r) \cdot B_{g_r,j}) \tag{2}$$
$$+(1 - C_{rj}) \cdot ((1 - w_r) + (2 \cdot w_r - 1) \cdot B_{g_r,j}))$$
$$+C_{k+1,j} \cdot (1 - B_{g_{k+1},j}) + (1 - C_{k+1,j}) \cdot B_{g_{k+1},j}$$
$$< (I(w_1, w_2, ..., w_k) + 1) \cdot (k + 1)$$

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

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

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. Using the R_{ij} variables, we can also weigh 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 $log_2(N-1)$, bits and so on with $log_2(N-i)$ 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, a truth table with 3 regulators approximately 8 bits and so on, and therefore we add 2^i to the weight of a gene's V_i variable, and apply telescopic cancellation as previously discussed. 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.

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 [19], [20]. 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 , then 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 is obtained, but this can result in a poor approximation of the optimal solution.

Results

Simulation

In order to compare the algorithm to other Boolean network inference methods, we generated 100 random Boolean networks and time series data using the BoolNet R package [23]. 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 [4] was used to solve using our method and the BESTFIT [**?**] method in the BoolNet package was used as comparison. The implementation of the REVEAL [22] method in the same package did not support noisy data. Figure 1 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.

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. [10] and the midbrain dopaminergic (mDA) neuron developmental GRN that was described by [2]. To obtain the gene counts we used the scRNA R package [13]. The R package Seurat [14] provides functions to inspect the data and determine the threshold for screening out cells with an unusually high or low number of features, leading to lower and upper thresholds of 500 and 4,000 features for this dataset, respectively. After filtering, the dataset contained 1,631 cells (experiments). Since scRNA-Seq data contains a high number of 0 counts, we applied SAVER [6] to impute expression values for the network genes. To obtain Boolean values, we 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 [4]. We then generated an EBNF description of the network and used the R package BoolNet to analyze the network properties [23]. For comparison to the inferred network, we generated 1,000 random networks with the same number of genes and the same mean number of inputs of a regulation function, using the 'generateRandomNKNetwork' function with the parameter 'topology' set to 'homogeneous'. Only 17 out of 1,000 networks had an equal or greater number of steady states than the inferred network, corresponding to a p-value of 0.017. We interpret this results 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. 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. [2]. 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. 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 4. 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.

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 heurisitic 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 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.

References

[1] Hirotogu Akaike. Information theory and an extension of the maximum likelihood principle. In *Springer Series in Statistics*, pages 199–213. Springer New York, 1998.

[2] Ernest Arenas, Mark Denham, and J. Carlos Villaescusa. How to make a midbrain dopaminergic neuron. *Development*, 142(11):1918–1936, June 2015.

[3] L. Geistlinger, G. Csaba, S. Dirmeier, R. Kuffner, and R. Zimmer. A comprehensive gene regulatory network for the diauxic shift in saccharomyces cerevisiae. *Nucleic Acids Research*, 41(18):8452–8463, July 2013.

[4] LLC Gurobi Optimization. Gurobi optimizer reference manual, 2021.

[5] R. F. Hashimoto, S. Kim, I. Shmulevich, W. Zhang, M. L. Bittner, and E. R. Dougherty. Growing genetic regulatory networks from seed genes. *Bioinformatics*, 20(8):1241–1247, February 2004.

[6] Mo Huang, Jingshu Wang, Eduardo Torre, Hannah Dueck, Sydney Shaffer, Roberto Bonasio, John I Murray, Arjun Raj, Mingyao Li, and Nancy R Zhang. Saver: gene expression recovery for single-cell rna sequencing. *Nature Methods*, 15(7):539–542, 2018.

[7] Guy Karlebach and Ron Shamir. Modelling and analysis of gene regulatory networks. *Nature Reviews Molecular Cell Biology*, 9(10):770–780, September 2008.

[8] Guy Karlebach and Ron Shamir. 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.

[9] S.A. Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, 22(3):437–467, March 1969.

[10] Gioele La Manno, Daniel Gyllborg, Simone Codeluppi, Kaneyasu Nishimura, Carmen Salto, Amit Zeisel, Lars E. Borm, Simon R.W. Stott, Enrique M. Toledo, J. Carlos Villaescusa, Peter Lönnerberg, Jesper Ryge, Roger A. Barker, Ernest Arenas, and Sten Linnarsson. Molecular diversity of midbrain development in mouse, human, and stem cells. *Cell*, 167(2):566–580.e19, October 2016.

[11] Takayuki Ohara, Timothy J. Hearn, Alex A.R. Webb, and Akiko Satake. Gene regulatory network models in response to sugars in the plant circadian system. *Journal of Theoretical Biology*, 457:137–151, November 2018.

[12] Jorma Rissanen. A universal prior for integers and estimation by minimum description length. *The Annals of Statistics*, 11(2), June 1983.

[13] Davide Risso and Michael Cole. *scRNAseq: Collection of Public Single-Cell RNA-Seq Datasets*, 2020. R package version 2.4.0.

[14] Rahul Satija, Jeffrey A Farrell, David Gennert, Alexander F Schier, and Aviv Regev. Spatial reconstruction of single-cell gene expression data. *Nature Biotechnology*, 33:495–502, 2015.

[15] Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2), March 1978.

[16] Roded Sharan and Richard M. Karp. Reconstructing boolean models of signaling. *Journal of Computational Biology*, 20(3):249–257, March 2013.

[17] Yoli Shavit, Boyan Yordanov, Sara-Jane Dunn, Christoph M. Wintersteiger, Tomoki Otani, Youssef Hamadi, Frederick J. Livesey, and Hillel Kugler. Automated synthesis and analysis of switching gene regulatory networks. *Biosystems*, 146:26–34, August 2016.

[18] Simon N Willis and Stephen L Nutt. New players in the gene regulatory network controlling late b cell differentiation. *Current Opinion in Immunology*, 58:68–74, June 2019.

[19] Soumita Seth , Saurav Mallik , Tapas Bhadra , Zhongming Zhao Dimensionality Reduction and Louvain Agglomerative Hierarchical Clustering for Cluster-Specified Frequent Biomarker Discovery in Single-Cell Sequencing Data *Frontiers in Genetics*, 13,2022

[20] Zehua Liu , Huazhe Lou , Kaikun Xie , Hao Wang , Ning Chen , Oscar M. Aparicio , Michael Q. Zhang , Rui Jiang , Ting Chen Reconstructing cell cycle pseudo time-series via single-cell transcriptome data *Nature Communications* 8:1,2017

[21] Laehdesmaeki2003) H. Laehdesmaeki, I. Shmulevich and O. Yli-Harja On Learning Gene-Regulatory Networks Under the Boolean Network Model Machine Learning 52:147–167 ,2003

[22] S. Liang, S. Fuhrman and R. Somogyi REVEAL, a general reverse engineering algorithm for inference of genetic network architectures Pacific Symposium on Biocomputing 3:18–29,1998

[23] Christoph Mussel, Martin Hopfensitz and Hans A. Kestler BoolNet – an R package for generation, reconstruction and analysis of Boolean networks Bioinformatics 26(10):1378-1380, 2010

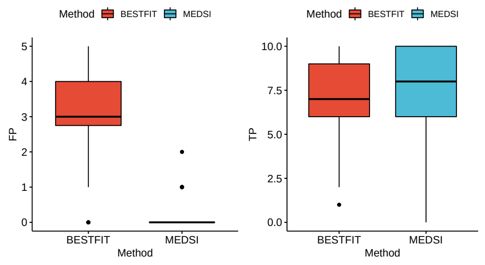

Figure 1. Boxplots of True Positives and False Positives

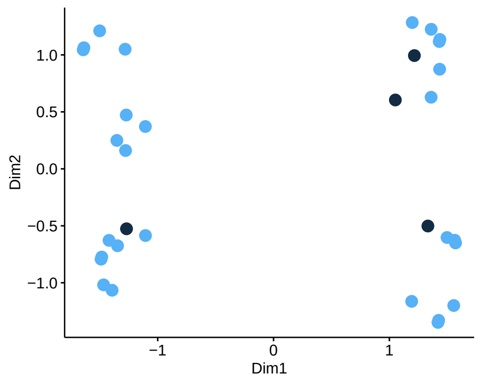

Figure 2. Multidimensional Scaling of Observed and Unobserved Network States

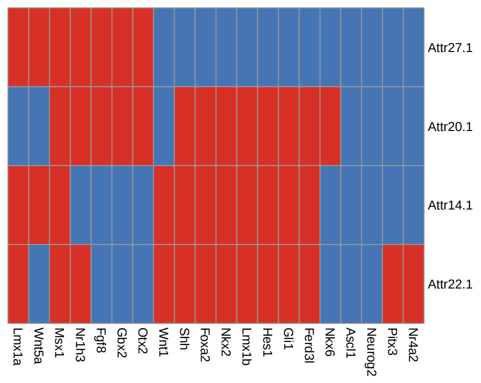

Figure 3. Clustering of Inferred Network States

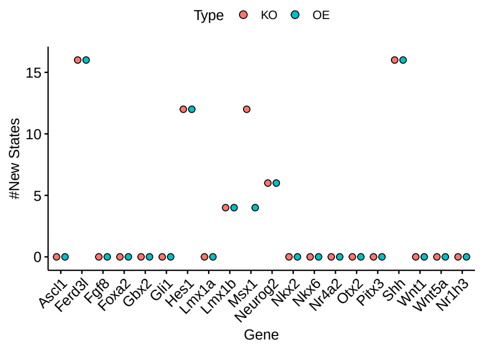

Figure 4. The Number of New Network States after Single Gene Overexpression and Knockout