Gene Regulatory Network Structure Prediction

Jacob Rast
jrast@andrew.cmu.edu

Leandro Lopez
lslopez@andrew.cmu.edu

Abstract

Biological systems and processes are networks of complex nonlinear regulatory interactions between nucleic acids, proteins, and metabolites. A natural way in which to represent these interaction networks is through the use of a graph. In this formulation, each node represents a nucleic acid, protein, or metabolite and edges represent intermolecular interactions (inhibition, regulation, promotion, coexpression, etc.). In this work, we present novel algorithms for the discovery of latent graph structures given experimental data.

1 Introduction

The problem of representing a biological process of interest in a graphical structure is under active investigation and stands as one of the grand challenges in biology.

Since RNA data is widely available, exclusive use of the transcriptome as a stand-in for other biomolecular expression levels is a well-studied formulation of the problem. Commonly, RNA microarray or single cell RNA-seq data is used to obtain gene expression levels of a large number of cells. This data is used to form a module of gene expression, expressed in a directed or undirected graph. If the "ground truth" or "gold standard" pathway is known, it can in turn be used to evaluate the network.

While the human body contains hundreds of cell types and subtypes, each cell contains a nearly identical genome. It is the complex regulatory machinery that determines which proteins a cell expresses and at what time, giving rise to cellular diversity. Regulatory networks play an important role in determining whether or not a gene will be expressed. Canonical examples of the regulatory network include the lactose (or lac) operon and Wnt pathway. The lac operon is a famous small regulatory network discovered in 1965 (7). An inhibitory transcription factor is typically expressed at high level, preventing the expression of genes responsible for the metabolism of lactose in bacteria. However, in the presence of lactose small co-inducer molecules will bind to the inhibitory protein allowing for the expression of three lac genes. Another famous network is the Wnt signaling pathway, which is involved in cancer and development (10).

Discovery of gene regulatory networks is a longstanding biological problem (13). The generation of high-throughput data relevant to the problem has allowed for the application of new approaches. A large influence in the field was a series of challenges issued from the National Center for Data to Health under the Dialogue on Reverse Engineering Assessment and Methods (DREAM) framework. DREAM challenges DREAM3 (19), DREAM4 (5) and DREAM5 (13) tasked participants with determining the network structures of a number of measurements of gene expression level from increasingly complex networks. These included both in simulated networks and those from well studied model organisms.

In the years since, a number of graphical methods have been developed expanding upon these approaches (17) (11). One notable example is the use of a factor graph for the representation of gene regulatory networks, which was successful at recovering pathways found in E. coli from a well studied database. Another notable example was the use of reciprocal graphs, a highly general structure that is

10708 Probabilistic Graphical Models Final Report. Spring 2022, Carnegie Mellon University

Figure 1: Full gene regulatory network for model organism E. coli

well suited to the study of gene expression given its ability to model loops, a drawback of traditional Bayesian networks.

Prediction of the regulatory structure of biological pathways has wide-ranging applications such as the discovery of new cellular pathways, prediction of response to environmental changes, discovery of transcription factors for stem cell differentiation, and deeper understanding of cellular pathways.

2 Background

2.1 Methods for modifying gene expression

Several tools exist for manipulating the gene expression level in a target organism or cell. One of such techniques is the gene knockout, whereby the expression level of a target gene or set of genes is forced to 0. Typically this is achieved through genome editing whereby the DNA sequence or sequences coding for the set of genes to be knocked out is removed. This renders the organism incapable of expressing the RNA or protein encoded. While most commonly a single gene or dual gene knockout is performed, some studies have identified genome screens that allow for the study of combinations of up to three genes to be knocked out (24). A number of methods have been developed for gene engineering and knockout. Briefly, technologies such as CRISPR/Cas9 (12) allow for the removal or insertion of a gene at any target location with minimal cost. An alternative approach for manipulation of gene expression level in a target cell is the gene knockdown. In some instances it is advantageous to study the effect of perturbing a gene away from its steady state expression level rather than silencing its expression entirely. This can be achieved through the introduction of inhibitory molecules such as interfering RNA (RNAi), antisense DNA oligos, or inhibitory proteins. Finally, some methods exists for inducing the expression of a target gene. While less robust and less well studied, some RNA species have been discovered that promote translation (2). In short, researchers have the tools to eliminate, increase, or decrease gene expression in a cell or organism. In this work algorithms were designed to study the experimental data resulting from each of these manipulations.

2.2 Mathematical models of gene expression

The use of tools to simulate gene expression data is invaluable to study the theory of reconstructing gene regulatory networks. A well-know tool for the simulation of gene expression is GeneNetWeaver (21). Briefly, GeneNetWeaver models a cell as a bipartite directed graph of proteins (transcription factors) and RNA. A series of ordinary differential equations or stochastic differential equations are used to relate the expression levels of the protein or RNA given a set of conditions. These conditions include biological factors such as RNA degradation rate, transcription factor-RNA affinity, transcription factor role (activation or deactivation of RNA transcription), etc. Interested readers may consult (8) and (21).

Steady-state solutions to the ODEs are used to generate baseline gene expression level. Experiments such as single gene knockout, multiple gene knockout, gene knockdown, gene knockup are simulated by intervening on the value of a set of variables and calculating the expression of the remaining variables under those conditions.

2.3 Boolean Networks

It is important to note the types of nonlinear dynamics that are frequently encountered in gene regulatory networks and that can be captured in ODE or SDE models. Frequently, proteins known as transcription factors will act in complexes in order to activate or inhibit the expression of a gene. The behavior whereby gene expression will only be affected by a complex and not at all by any subset of the complex is similar to the Boolean AND function. Simple Boolean network models of gene regulatory networks have been applied to capture this behavior (1). For more expressive power and the creation of synthetic datasets, success has been reported converting a Boolean network to a system of ordinary differential equations (18).

3 Related work

3.1 Notation and Notable Approaches

In our first experimental setup we formulate the Gene Regulatory Network (GRN) prediction problem as follows: let $\mathcal{G} = \{1 \dots p\}$ represent a universe of genes with $\mathcal{T} \subset \mathcal{G}$ transcription factors. We wish to discover interactions of the form $\mathcal{E} = \{(t, g) \mid t \in \mathcal{T}, g \in \mathcal{G}\}$, where $\mathcal{T}$ is known a priori. In context, an edge between t and g implies that the protein product(s) of transcription factor t has an effect on the expression of gene g. For large graphs, exhaustively searching $\mathcal{E}$ is intractable. To approximate true regulatory relationships $\mathcal{E}^* \subset \mathcal{E}$, we take as input a single cell gene expression data set which is simply a data matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ where $\mathbf{X}_{ij}$ represents the expression level of gene $j \in \mathcal{G}$ in experimental condition $i \in [n]$ for a single cell.

More recent approaches interpret GRN prediction as a feature selection problem. Given some $g \in \mathcal{G}$, regress the expression of g on the expression of all transcription factors $\mathcal{T}$

$$X_g = f_g(X_{\mathcal{T}}) + \epsilon$$

Functional forms for f_g have ranged from random forests (6), boosted decision trees (14), to linear regression with a Lasso penalty (13). After estimating f_g for all genes, these methods rank the strength of each (t, g) based on the estimated f_g. One drawback of this approach is whether f_g is expressive enough to capture complex networks found in nature. Another drawback is the scoring required after the estimation of f_g may have to be adjusted for different networks, making it an ad-hoc. Additionally, explicit regularization of f_g may not capture desired topological priors of the underlying GRN.

Alternatively, we can fix some $g \in \mathcal{G}$ and create an undirected sub-network consisting of a single target g along with all candidate transcription factors $\mathcal{T}$. The structure of this sub-network can be learned using $\mathbf{X}$. A flexible model for this sub-problem is Gaussian Copulas (15), which reduce our problem to estimating a suitable precision matrix. This approach has several tractable estimation methods using MCMC techniques (16). The sampling methodology can be adjusted to encourage sparse edges between target g and transcription factors $\mathcal{T}$. Some drawbacks of this methodology are the convergence of MCMC to a suitable posterior and intractability for large network sizes.

4 Methods

4.1 GRNULAR + TopoDiffVAE

4.1.1 GRNULAR Framework

Letting f_g be a neural network gives us greater expressive power than the above approaches, while introducing new challenges. One challenge is the limited amount of labelled data for supervision (particularly for large GRNs). Fortunately, there has been an improvement in simulating single-cell gene expression experiments enabling the training of deep architectures on synthetic datasets. For our study we use the SERGIO simulator(4).

Suitable architectural choices for graph estimation require the number of parameters grow sub-quadratically with the number of genes since both the input and the output are matrices(22). We take an algorithm unrolling approach based on GRNULAR (23) which starts from a fully connected NN $f_{\mathcal{W}}$ which given expression levels $X_{\mathcal{T}}$ outputs expression levels $X_{\mathcal{G}}$. It has weights $\mathcal{W} = \{W_1, W_2, \ldots, W_c\}$ and we iteratively zero out the edge weights at each iteration. At the end of the algorithm, $(t, g) \in \mathcal{E}^*$ iff there exists a path from X_t to X_g in $f_{\mathcal{W}}$. Mathematically, we recover this dependence $\forall (t, g) \in \mathcal{E}$ by a matrix product $\Theta = \prod_i |W_i|$. To achieve sparsity, we penalize the MSE error with an l_1 penalty term, and introduce a Lagrangian variable $Z = \prod_i |W_i|$ as in (23). Minimizing Z, Θ consecutively for $l \in [L-1]$ iterations we have

$$\mathcal{W}^{(l+1)} \leftarrow \arg\min_{\mathcal{W}} \sum_{k=1}^{N} \left\| X_{\mathcal{G}}^{k} - f_{\mathcal{W}} \right\|^2 + \frac{1}{2}\lambda \left\| \prod_i |W_i| - Z^l \right\|_F^2 \tag{1}$$

$$Z^{(l+1)} \leftarrow \eta_\rho \left(\prod_i |W_i^{(l+1)}| \right) \tag{2}$$

Where η is a proximity operator given by $\eta_\rho(\theta) = sign(\theta)\max(|\theta| - \rho, 0)$, thus allowing our optimization to descend more stably. In addition to $\mathcal{W}$, we also update entry-wise thresholding operator ρ and step size λ at each iteration as follows.

$$\lambda^{(l+1)} \leftarrow \Lambda_{nn}(\|Z - \Theta\|_F^2,\ \lambda^l) \tag{3}$$

$$\rho^{(l+1)} \leftarrow \rho_{nn}(\Theta,\ \hat{\Sigma}_{\mathcal{T}},\ Z) \tag{4}$$

Where ρ_{nn}, λ_{nn} are fully connected shallow NNs that use the solution of the previous update to generate the next one, and $\hat{\Sigma}_{\mathcal{T}} \subset \Sigma$ is the covariance sub-matrix of $\mathbf{X}$ using only transcription factors $\mathcal{T}$. We also generate a "good" initialization $\mathcal{W}^0$ by solely optimizing on the first term of equation 1 before we begin unrolling the iterations. Thus far this is identical to GRNULAR (23) (see Figure 2 in this reference for a schematic representation).

4.1.2 Topological Difference Variational Autoencoder (TopoDiffVAE)

The proximal operator η, can be thought of projecting Θ onto the space of graphs sharing the same topological properties as the target GRN Θ^*. At the last layer of unrolling L we substitute η with a topological difference variational autoencoder (TopoDiffVAE) (20) to adaptively learn topological priors from the training data, thus expanding the expressibility of $f_{\mathcal{W}}$. As in (20), we consider a conditional VAE such that $\mathcal{P} : (Z^L, \mathbf{s}) \rightarrow Z^{L+1}$, where latent variable $\mathbf{s}$ captures the topological difference between Z^L and Θ^*. We proceed very similarly to (20).

Encoder. Let $\mathbf{A_Z}$ be the binary adjacency matrix of Z^l, and $\mathbf{A_{\Theta^*}}$ the binary adjacency matrix for Θ^*. The approximate the posterior $q(\mathbf{s}|Z^l, \Theta^*)$ is a Gaussian whose mean μ and covariance $\mathbf{\Sigma} = \sigma\mathbf{I}$ are transformed from the embedding $f_{\text{emb}}(\mathbf{A_Z})$ and $f_{\text{emb}}(\mathbf{A_{\Theta^*}})$ as follows

$$\delta = f_{\text{emb}}(\mathbf{A_{\Theta^*}}) - f_{\text{emb}}(\mathbf{A_Z}) \tag{5}$$

$$\mu = f_{\text{mean}}(\delta), \quad \sigma = f_{\text{cov}}(\delta), \quad \mathbf{s}|Z^L, \Theta^* \sim \mathcal{N}(\mu, \sigma) \tag{6}$$

where f_{mean} and $f_{\text{cov}}(\delta)$ are 2-layer fully connected NNs, and f_{emb} is a 2-layer graph convolutional network (9) followed by readout layer f_{readout} that averages the node representation to obtain a graph representation,

$$f_{\text{emb}} = f_{\text{readout}}\Big(\psi(\mathbf{A}^T\psi(\mathbf{A}\mathbf{d}\mathbf{h}_0^T)\mathbf{H_1})\Big) \tag{7}$$

Where $\mathbf{A}$ refers to $\mathbf{A_Z}$ or $\mathbf{A}_{\mathbf{\Theta}^*}$, and $\mathbf{h}_0$, $\mathbf{H_1}$ are learnable parameters, ψ is an activation function, and $\mathbf{d} = \mathbf{A1}$ represents the node degrees of transcription factors $\mathcal{T}$. We sample use the Gaussian reparametrization trick to sample from our latent variable $\mathbf{s}$.

Decoder. We obtain an augmented graph representation by concatenating Z^l and $\mathbf{s}$ and then feed it into a 2-layer fully connected NN

$$Z^{(L+1)} = f_{\text{desc}}(\text{CONCAT}[Z^l, \mathbf{s}]) \tag{8}$$

4.1.3 End-to-End Training Scheme

By stacking TopoDiffVAE on top of GRNULAR, we augment the existing method by directly learning a data-to-graph mapping that minimizes the following loss:

$$J = \sum_{l=0}^{L}\Big\{\tau^{L-l}\frac{\|Z^l - \Theta^*\|^2}{\|\Theta^*\|_2^2}\Big\} + \beta_{KL}KL\Big(q(\mathbf{s}|Z^{L+1}, \Theta^*)\|\mathcal{N}(\mathbf{0}, \mathbf{I})\Big) \tag{9}$$

Where $\tau \in (0, 1]$ is a loss-discounting factor and β_{KL} is a trade-off hyperparameter.

4.2 Boolean Causal Discovery Algorithm

In this section, the development, motivation, working theory, and proof of the Bool-PC algorithm are described.

4.2.1 Test for independence

Using observational data alone it is theoretically possible test for independence between variables using an approach such as mutual information. For small or simple networks without nonlinearities this approach can successfully reconstruct a directed graph (3). However, for larger or more complex networks this approach yields poor performance. Recognizing that the ability to manipulate the expression level of a gene is equivalent to intervening on a variable (in the causal sense) a much more robust test for independence can be developed.

We use the property that for two independent distributions, $P(A, B) = P(A)P(B)$ to write the following:

$$A \perp\!\!\!\perp B \implies P(A = a|B = b_1) = P(A = a|B = 0) \tag{10}$$

In other words, if the probability distribution of A is unchanged by knocking out gene B, A and B are independent.

Additionally, note that for a directed network we have $A \perp\!\!\!\perp B \not\Longrightarrow B \perp\!\!\!\perp A$ for independence as defined in this equation 10.

4.2.2 Test for Conditional Independence

This property is now extended to develop a test for conditional independence. A naïve extension would propose the following:

$$A \perp\!\!\!\perp B|C \implies P(A = a|B = b_1, C = c_1) = P(A = a|B = 0, C = c_1) \tag{11}$$

Equation 11 holds for many probabilistic graphical models. It follows from the definition of conditional independence $P(A|B, C) = P(A|C)$.

Importantly, equation 11 does not hold for the problem of gene regulatory network inference containing Boolean nonlinearitites. An illustrative counterexample can be found in the toy network depicted

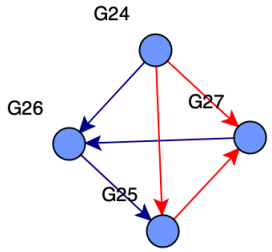

Figure 2: Toy model of conditional dependence and independence

in figure 2. From the graph structure, we can read the set of conditional independence statements $G27 \perp\!\!\!\perp G26|G25$ and $G27 \not\perp\!\!\!\perp G24|G25$.

In table 1 we consider the effect of perturbing the value of G26 given G25 = 0. The value of G27 is unaffected, seemingly in accord with the conditional independence statements given in 11. However, in table 2 we consider the effect of perturbing G24 given G25 = 0. Under the same set of experimental conditions we observe a nearly identical result. The value of G27 is unaffected given G25=0, despite the graph structure giving conditional dependence. Finally, in 3 the effect of perturbing G25 given G24 = 0 is given. Notice that again, no effect is observed on G27 from perturbations in G25 given G24 = 0, despite the graph structure describing conditional dependence.

Given the Boolean AND inhibitory effect of G24 and G25 on G27, equation 11 does not capture the graph structure. This motivates the development of a test for conditional independence in Boolean causal graphs, given in 12.

4.3 Bool-PC

Equation 12 is now used to develop a variant of the PC algorithm. Algorithm Bool-PC (1) is for use in Boolean-Nonlinear causal graphs suitable for use in gene regulatory network discovery.

Intuitively, the algorithm has the following logic. For any sets of variables X, Y, and Z if there is some flow of causality from X to Y, as shown in Step 1, we cannot prune an edge given Z blocks causal from X to Y if Y also blocks causal flow from Z to X, as these two conditions would imply no causal flow from Y, Z to X.

4.4 Bool-PC Proof of Correctness

The proof of correctness for Bool-PC follows from the reduction of SAT to 3SAT. For any 3 variables X, Y, Z we have shown exhaustively that Bool-PC can discover any graph structure, as shown in the toy example from Figure 2. Given that the algorithm can reconstruct the Boolean relations between any 3 variables and that sets of 3 variables can be used to satisfy the Boolean relation of any arbitrary Boolean Satisfiability problem, we argue that Bool-PC is a correct algorithm for any arbitrary Boolean graph. From here the correctness of the PC algorithm applies.

$$A \perp\!\!\!\perp B|C \rightarrow P(A = a|B = b_1, C = c_1) = P(A = a|B = 0, C = c_1) \\ \wedge P(A = a|B = b_1, C = c_1) = P(A = a|B = b_1, C = 0) \quad (12)$$

5 Results

Performance of the GRNULAR + TopoDiffVAE algorithm, Bool-PC algorithm and comparison against the state of the art can be found in table 4.

Algorithm 1 Bool-PC Algorithm

for `all edges` $E_{X,Y}$ **do** ▷ Step 1
  if $P(X) = P(X|Y)$ **then**
    `Remove edge` $E_{X,Y}$
  end if
  for `Remaining edges` $E_{X,Y}$ `and sets of third variable Z` **do** ▷ Step 2 to N
    if $P(X|Y,Z) = P(X|Y)$ and $P(X|Y,Z) = P(X|Z)$ **then**
      `Remove edge` $E_{X,Y}$
    end if
  end for

Conditions	G24	G25	G26	G27
Steady state	0.541	0.464	0.110	0.112
G25 = 0	0.541	0.00	0.665	1.00
G25 = 0, G26=0	0.541	0.00	0.00	1.00
G25 = 0, G26 ↑	0.541	0.00	1.99	1.00

Table 1: Experimental perturbations of G25 and G27

Conditions	G24	G25	G26	G27
Steady state	0.541	0.464	0.110	0.112
G25 = 0	0.54	0.00	0.67	1.00
G25 = 0, G24 = 0	0.00	0.00	0.09	1.00
G25 = 0, G24 ↑	2.07	0.09	0.69	1.00

Table 2: Experimental perturbations of G25 and G24

Conditions	G24	G25	G26	G27
Steady state	0.541	0.464	0.110	0.112
G24 = 0	0.00	0.68	0.09	1.00
G24 = 0, G25 = 0	0.00	0.00	0.09	1.00
G24 = 0, G25 ↑	0.00	2.07	0.09	1.00

Table 3: Experimental perturbations of G24 and G25

Metrics	AUPRC	Training Time (seconds)
GRNULAR	0.640217	491.14
GRNULAR + TopoDiffVAE	0.894	478.95
Bool-PC (dual knockout)	0.741	NA
Bool-PC (n-knockout, theoretical)	1.00	NA

Table 4: Performance comparison of discovery algorithms on simulated GRN with 100 genes, and 10 Transcription factors in a noise-free setting

6 Discussion and Analysis

6.1 GRNULAR + TopoDiffVAE

One disadvantage of GRNULAR + TopoDiffVAE is increased memory usage during training. This prevented us from testing out larger GRN's. Additionally, we did not check the performance of GRNULAR + TopoDiffVAEunder on real biological data, which means that the increased performance may not generalize to realistic GRNs. Another limitation of this study is that the set of transcription factors $\mathcal{T}$ must be known a priori. Like any supervised method, we must be confident that the simulator approximates the topological properties of the biological GRN we are trying to model. Otherwise our VAE layer is overfitting to the simulator's dynamics instead of the target GRN. More testing also needs to be done under noisy observations of single-cell expression data.

6.2 Bool-PC

First, it must be noted that the results present in table 4 are for data generated without noise. The theoretically performance of Bool-PC on a network defined by a system of SDEs with noise (a more challenging problem that closer resembles the biological reality) cannot in general reach AUPRC of 1. Additionally, a major limitation of the Bool-PC algorithm is the experimental and time complexity. The PC algorithm has exponential time complexity, limiting its application to the discovery of full GRNs of thousands of nodes. Worse, the algorithm proposes an exponential number of gene knockout experiments which is both prohibitively expensive and technically infeasible. Bool-PC with dual gene knockout reliably captures well gene regulatory subnetworks of hundreds of nodes and very often perfectly captures networks of tens of nodes. This makes it a useful albiet limited tool. Use of the same conditional independence tests with more recent causal discovery algorithms such as the Grow-Shrink algorithm may expand the usefulness of the approach. Additionally, as gene sequencing costs shrink massive gene knockout panels will be increasingly feasible.

7 Appendix

7.1 Teammates and work division

7.1.1 Jacob Rast

Implementation and development of Bool-PC algorithm. Biological relevance.

7.1.2 Leandro Lopez

Implementation and development of GRNULAR + TopoDiffVAE.

7.2 Access to Code

All relevant data and code for the project can be found at the project's Github repository https://github.com/jacobrast/grn_causal_discovery.

References

[1] ALBERT, R., AND OTHMER, H. G. The topology of the regulatory interactions predicts the expression pattern of the segment polarity genes in drosophila melanogaster. *J. Theor. Biol. 223*, 1 (July 2003), 1–18.

[2] CARRIERI, C., CIMATTI, L., BIAGIOLI, M., BEUGNET, A., ZUCCHELLI, S., FEDELE, S., PESCE, E., FERRER, I., COLLAVIN, L., SANTORO, C., FORREST, A. R. R., CARNINCI, P., BIFFO, S., STUPKA, E., AND GUSTINCICH, S. Long non-coding antisense RNA controls uchl1 translation through an embedded SINEB2 repeat. *Nature 491*, 7424 (Nov. 2012), 454–457.

[3] CHOW, C., AND LIU, C. Approximating discrete probability distributions with dependence trees. *IEEE Trans. Inf. Theory 14*, 3 (May 1968), 462–467.

[4] DIBAEINIA, P., AND SINHA, S. Sergio: a single-cell expression simulator guided by gene regulatory networks. *Cell systems 11*, 3 (2020), 252–271.

[5] GREENFIELD, A., MADAR, A., OSTRER, H., AND BONNEAU, R. Dream4: Combining genetic and dynamic information to identify biological networks and dynamical models. *PloS one 5*, 10 (2010), e13397.

[6] HUYNH-THU, V. A., IRRTHUM, A., WEHENKEL, L., AND GEURTS, P. Inferring regulatory networks from expression data using tree-based methods. *PloS one 5*, 9 (2010), e12776.

[7] JACOB, F., AND MONOD, J. Genetic regulatory mechanisms in the synthesis of proteins. *Journal of molecular biology 3*, 3 (1961), 318–356.

[8] KANG, X., HAJEK, B., AND HANZAWA, Y. From graph topology to ODE models for gene regulatory networks. *PLoS One 15*, 6 (June 2020), e0235070.

[9] KIPF, T. N., AND WELLING, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[10] KOMIYA, Y., AND HABAS, R. Wnt signal transduction pathways. *Organogenesis 4*, 2 (2008), 68–75.

[11] KOTIANG, S., AND ESLAMI, A. A probabilistic graphical model for system-wide analysis of gene regulatory networks. *Bioinformatics 36*, 10 (2020), 3192–3199.

[12] MALI, P., YANG, L., ESVELT, K. M., AACH, J., GUELL, M., DICARLO, J. E., NORVILLE, J. E., AND CHURCH, G. M. Rna-guided human genome engineering via cas9. *Science 339*, 6121 (2013), 823–826.

[13] MARBACH, D., COSTELLO, J. C., KÜFFNER, R., VEGA, N. M., PRILL, R. J., CAMACHO, D. M., ALLISON, K. R., KELLIS, M., COLLINS, J. J., AND STOLOVITZKY, G. Wisdom of crowds for robust gene network inference. *Nature methods 9*, 8 (2012), 796–804.

[14] MOERMAN, T., SANTOS, S. A., GONZÁLEZ-BLAS, C. B., SIMM, J., MOREAU, Y., AERTS, J., AND AERTS, S. GRNBoost2 and arboreto: efficient and scalable inference of gene regulatory networks. *Bioinformatics 35*, 12 (Nov. 2018), 2159–2161.

[15] MOHAMMADI, A., ABEGAZ, F., VAN DEN HEUVEL, E., AND WIT, E. C. Bayesian modelling of dupuytren disease by using gaussian copula graphical models. *Journal of the Royal Statistical Society: Series C (Applied Statistics) 66*, 3 (2017), 629–645.

[16] MOHAMMADI, R., AND WIT, E. C. Bdgraph: An r package for bayesian structure learning in graphical models. *arXiv preprint arXiv:1501.05108* (2015).

[17] NI, Y., MÜLLER, P., WEI, L., AND JI, Y. Bayesian graphical models for computational network biology. *BMC bioinformatics 19*, 3 (2018), 59–69.

[18] PRATAPA, A., JALIHAL, A. P., LAW, J. N., BHARADWAJ, A., AND MURALI, T. M. Benchmarking algorithms for gene regulatory network inference from single-cell transcriptomic data. *Nat. Methods 17*, 2 (Feb. 2020), 147–154.

[19] PRILL, R. J., MARBACH, D., SAEZ-RODRIGUEZ, J., SORGER, P. K., ALEXOPOULOS, L. G., XUE, X., CLARKE, N. D., ALTAN-BONNET, G., AND STOLOVITZKY, G. Towards a rigorous assessment of systems biology models: the dream3 challenges. *PloS one 5*, 2 (2010), e9202.

[20] PU, X., CAO, T., ZHANG, X., DONG, X., AND CHEN, S. Learning to learn graph topologies. *Advances in Neural Information Processing Systems 34* (2021).

[21] SCHAFFTER, T., MARBACH, D., AND FLOREANO, D. GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods. *Bioinformatics 27*, 16 (Aug. 2011), 2263–2270.

[22] SHRIVASTAVA, H., CHEN, X., CHEN, B., LAN, G., ALURU, S., LIU, H., AND SONG, L. Glad: Learning sparse graph recovery. *arXiv preprint arXiv:1906.00271* (2019).

[23] SHRIVASTAVA, H., ZHANG, X., SONG, L., AND ALURU, S. Grnular: A deep learning framework for recovering single-cell gene regulatory networks. *Journal of Computational Biology 29*, 1 (2022), 27–44.

[24] ZHOU, P., CHAN, B. K., WAN, Y. K., YUEN, C. T., CHOI, G. C., LI, X., TONG, C. S., ZHONG, S. S., SUN, J., BAO, Y., MAK, S. Y., CHOW, M. Z., KHAW, J. V., LEUNG, S. Y., ZHENG, Z., CHEUNG, L. W., TAN, K., WONG, K. H., CHAN, H. E., AND WONG, A. S. A three-way combinatorial crispr screen for analyzing interactions among druggable targets. *Cell Reports 32*, 6 (2020), 108020.