Effective and Scalable Causal Inference from Gene Expression Time Series

Jonathan Lu

Advisor: Professor Barbara Engelhardt

Submitted in partial fulfillment

of the requirements for the degree of

Bachelor of Science in Engineering

Department of Computer Science

Princeton University

May 2018

I hereby declare that I am the sole author of this thesis.

I authorize Princeton University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

_____

Jonathan Lu

I further authorize Princeton University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

_____

Jonathan Lu

Abstract

Gene regulatory network inference holds great potential to uncover the biological mechanisms of disease and inform downstream experiments. However, effective inference of such causal relations is not straightforward. Methods for network inference must demonstrate accuracy to handle molecular interactions, statistical significance to decide on thresholds, and scalability to handle high-throughput sequencing assays. We introduce BETS (Bootstrap Elastic net regression from Time Series) to address these issues. BETS improves network inference in three ways: 1) it uses the elastic net regression penalty to handle correlated genes, 2) it ranks edges based on a new measure of their stability, the "bootstrap frequency", and 3) it is highly parallelized, allowing analysis of datasets of thousands of genes in only a few days. Through these three innovations, our method has ranked 3rd in AUROC (out of 17) and 6th in AUPR (out of 22) in the DREAM4 100-gene community benchmark for gene regulatory network inference. Importantly, our method is one of the fastest methods compared with methods of similar performance . We next run on the GR project data, consisting of 2768 differentially-expressed genes across 12 timepoints. We infer a causal network of 31,945 edges, which we evaluate on multiple sources of held-out data, including over-expression data from the same experimental setup and literature-curated regulatory relationships.

Acknowledgements

First and foremost, I wish to thank my primary mentors Bianca Dumitrascu and Professor Engelhardt for their continued guidance and support. Thank you for being so kind, encouraging and patient past all mistakes and roadblocks on my part.

I would also like to thank Professor Mona Singh for taking time out of her busy schedule to be the second reader for my thesis.

I would like to thank my other mentors, Brian Jo, Greg Darnell, Ian McDowell (Duke University), and Professor Tim Reddy (Duke University). Brian, thank you for being so helpful and encouraging through every stage of this project, and for volunteering your time to help us validate these results! Greg, thank you for so generously volunteering your time every time I come to ask you, whether it's on doing FDR calibration or estimating standard errors. You've been so thorough and helpful every time! Ian and Professor Reddy, thank you for your guidance in helping us to understand the biological context of the system and to capture the fascinating biology behind the GR dataset.

I wish to thank the other members of the Engelhardt lab, including Derek, Li-Fang, Ari, Izzy, and Allison. Whether it was explaining a concept during our paper reviews, running jobs on the cluster, or helping me solve a tricky Linux problem, I am so grateful to you for always being there to help!

I would like to thank Dr. Christopher Penfold (Cambridge University, United Kingdom) and Dr. Vân Anh Huynh-Thu (University of Liege, Belgium) for sharing their scripts and explaining to me their gene regulatory network inference methods, CSI and Jump3, respectively. I have learned a lot from you both!

Finally, I would like to thank the data collection team at the Reddy Lab (Duke University) for gathering and sharing the fascinating glucocorticoid receptor dataset.

To my parents, who just keep believing in me.

Contents

Chapter 1

Introduction

Causal networks are a fundamental model for understanding how cells develop and respond to the environment as dynamic entities [15, 24]. The primary unit of control within a cell is the protein. Translated from gene transcripts, proteins control everything from metabolic breakdown to hormone release to immune system alerts. Proteins also interact (activate or repress) with each other's production at the gene level. These interactions form a complex causal regulatory network [24]. However, direct protein levels are difficult to assay [24].

Gene regulatory networks provide an approximate description for the causal regulatory dynamics between proteins. Gene transcript levels, easily measured by next generation sequencing technologies, can be used as a proxy for protein levels [56, 24]. Network inference methods vary widely in both framework and effectiveness. Yet, even if they are only marginally effective, they can still be useful for biologists: they help biologists narrow the subset of genes and interactions to follow up on with direct experimentation. In this thesis, we study the problem of gene regulatory network inference from gene expression time series.

We are specifically interested in understanding the glucocorticoid genomic response. Glucocorticoids are used to control overactive immune reactions [45, 46, 7, 31].

However, they lead to metabolic side effects, such as diabetes and obesity [11, 51]. We aim to understand the gene regulatory network behind the glucocorticoid response, with the ultimate goal of designing improved, targeted drugs for different pathways. We use gene expression time series data collected from the Glucocorticoid Receptor (GR) project; we shall refer to this as the GR data.

From a data science perspective, the problem of regulatory network inference from time series is difficult for two main reasons: causal inference and high dimensionality. Inference of a gene X causing a gene Y based on time series is challenging because there are multiple ways the regulatory relationship could appear: it could, for example, be nonlinear or nonstationary. High dimensionality (i.e. when the number of predictors, or genes, exceeds the number of samples) is common among gene expression assays because there are about 18000 protein-coding genes. The dimensionality adds several challenges: first, many well-established statistical tests fail in the high-dimensional setting, and it is difficult to account for all possible effects from the other genes in a relationship. Moreover, methods must scale to the high dimension of thousands or even tens of thousands of genes.

In this work, we develop BETS (Bootstrap Elastic net inference from Time Series), a form of vector autoregression (VAR) that addresses the major challenges of regulatory network inference. We build off of the VAR framework due to its simplicity, speed, and empirical effectiveness [25, 57, 27]. **BETS' key contribution is to use stability to rank edges as opposed to coefficients**. That is, we are more confident in an edge's validity if it is stable to small perturbations to the data (in the form of bootstrap samples).

BETS is effective at causal inference, provides statistical significance, and scales to large datasets. BETS successfully infers causality from high-dimensional gene expression time series, being the top-performing method of its class and competitive with other top methods on the DREAM4 100-gene Network Infer-

ence Challenge [29, 38]. It is faster than many methods of similar performance due to its use of parallelization and the elastic net penaltyl. We benchmark BETS against methods from a literature review, as well as in-house runs of 5 other methods based on mutual information, decision trees, and Gaussian processes. We then use BETS to infer and analyze 2 networks from the Glucocorticoid Receptor (GR) project. We analyze the bootstrap procedure to understand the correspondence of the original fits with the bootstrap fits. Finally, we evaluate our inferred network on sources of external data, including an over-expression dataset and a literature-curated network.

Chapter 2

Background and Related Work

2.1 Gene Regulatory Network Inference Methods

Much of this literature review is based on a similar review performed in my junior thesis work [27].

A variety of approaches for causal inference in gene regulatory networks have been studied over the past decade. We review the approaches and discuss several comparison studies. A more detailed treatment can be found in [24].

- **mutual information:** Compute the mutual information between genes' expression profiles to declare causal edges.
- **vector autoregression**: Regress one gene's expression values as a (usually linear) function of other genes' previous expression values
- **ordinary differential equation**: Regress one gene's expression's derivative as a (usually linear) function of other genes' current expression levels
- **dynamic Bayesian network**: Perform a causal graph search to find the graph with maximizal posterior probability

- **decision tree**: Fit one gene's state as a decision tree function of other gene's expression levels

- **Gaussian process**: Fit one gene's state as a function, from a Gaussian process distribution, of other gene's expression levels

2.1.1 Mutual Information

Mutual information (MI) methods compute the mutual information between the expression values of genes X and Y. In the time series setup, for a given lag k, the causal gene's lagged values are used with the effect gene's current values [28, 25].

$$I^k(X,Y) = \sum_{t=k}^{T} P(X_{t-k}, Y_t) \log \frac{P(X_{t-k}, Y_t)}{P(X_{t-k})P(Y_t)} \tag{2.1}$$

Methods such as Context-likelihood of Relatedness (CLR) [10], Maximum Relevance-Minimum Redundancy Network (MRNET) [33], and ARACNE [30] find gene network edges using the mutual information between lagged values; the key difference is how each accounts for possible indirect interactions and background correlations. CLR computes a background distribution of MI values involving X as a cause or Y as an effect, and then declares significant edges based on that background distribution. MRNET greedily searches possible edges using the maximum relevance-minimum redundancy principle. It selects causes that maximize the relevance (mutual information with the effect) minus the redundancy (mutual information with present causes). ARACNE computes all pairwise mutual information values and eliminates direct edges in triplets.

MI methods have the advantage of being simple and fast; this year, tl-CLR, a time-lagged version of CLR, was one of the top performers on the DREAM4 100-gene challenge [12], surpassing historically poor performance by these methods [38]. However, mutual information does not give immediate insight into the sign (i.e. activation

or repression) of genes' relation because it is a nonnegative metric. [33, 61]

2.1.2 Vector Autoregression

Vector autoregression (VAR) fits a gene's expression as a (often linear) function of the lagged expression values of other genes, the "predictors" [53]. "Granger Causality" methods, in which $X \to Y$ if including X in our predictors significantly improves our ability to predict Y, tend to fall into the VAR framework [13]. For a given lag K and genes X and Y, VAR fits:

$$Y_t = \sum_{i=1}^{K} \alpha_i Y_{t-1} + \sum_{g \in G} \sum_{i=1}^{K} \beta_i^g X_{t-i}^g + \epsilon_t \tag{2.2}$$

Older VAR analyses used pairwise fits and parametrized statistical tests to detect edges [35, 53, 60]. However, the high number of possible predictors in genome-wide sequencing assays poses challenges to these analyses, requiring these functions to be fit using regularization techniques such as LASSO [26, 50], James-Stein Shrinkage [36], or Ridge regression [57]. Nonlinear, kernel-valued functions have also been used for the regression [23]. Only 3 VAR methods: LASSO, GCCA, and OKVAR-Boost, have been tested on the DREAM4 data [38, 12]. We develop BETS, the top-performing vector autoregression method on the DREAM4 100-gene challenge. The performance results from our novel combination of the elastic net penalty and bootstrap fits for stability selection.

2.1.3 Ordinary Differential Equation

Ordinary differential equations (ODE) fit the derivative of a gene Y's expression as a function of all current gene expression values $Y_t, X_t^1, \ldots, X_t^G$ [24, 3, 5].

$$\frac{dY_t}{dt} = f(Y_t, X_t^1, \ldots X_t^G) + \epsilon_t \tag{2.3}$$

Time-Series Network Identification (TSNI) used a linear ODE with spline interpolation to gain more timepoints [3]. Inferelator used a special piecewise linear "g" function in their ODE [5]. The function used in an ODE can also be adapted to be more complex and nonlinear, for example using S functions or polynomials, but the parameterized types often have too many parameters to be estimated accurately from the limited data [15].

2.1.4 Dynamic Bayesian Network

Dynamic Bayesian networks (DBN) search the space of possible causal graph structures and identify the structure with the highest posterior probability given the data [21, 14, 58]. Where *pa* denotes the parents, i.e. causes, of a given gene node:

$$P(X_{1:T}^{1:n}) = \prod_{t=1}^{T} \prod_{i=1}^{n} P(X_t^i | pa(X_t^i)) \tag{2.4}$$

G1DBN approximates the true causal graph using low-order conditional dependencies [21]. BANJO proposes new graphs, checks for cycles, computes the score of the network, and decides whether to accept it [14]. Many methods have also used hidden states [4, 43]. While they have been shown to be effective on smaller datasets [62], they scale very poorly due to the superexponential growth of possible causal graph structures [39, 57]. The one exception is ScanBMA, which uses a pruning method based on Occam's window to limit the search space and gain a speedup. However, it has been less effective than approaches that are not based on dynamic Bayesian networks [58].

2.1.5 Decision Tree

Decision trees can handle more complex relationships between genes than linear functions. Thus, they have been used to improve upon linear methods in both the vector

autoregression and ordinary differential equation settings.

For a gene Y, either its expression Y_t or the derivative of its expression $\frac{dY_t}{dt}$, respectively, is modeled as a decision tree function (DT) of previous expression values [19, 12].

$$Y_t = f(Y_{t-i}, X^g_{t-i}) + \epsilon_t, f \sim DT \tag{2.5}$$

$$\frac{dY_t}{dt} = f(Y_{t-i}, X^g_{t-i}) + \epsilon_t, f \sim DT \tag{2.6}$$

Two recent methods, Jump3 [19] and dynGENIE3 [12], are among the top-performing methods on the DREAM4 100-gene Challenge. Jump3, which combines hidden promoter states and "jump trees", is relatively slower due to its scaling at $O(N^4)$; meanwhile, dynGENIE3, which uses ordinary differential equations and decision trees, has been one of the fastest methods, scaling to datasets of about 2000 genes in only 90 minutes. One limitation of decision tree methods is that they only produce a ranking of edges, without a framework for setting a significance threshold [19, 12]. Moreover, these rank edges based on their positive "feature importances" without accounting for the sign. Like mutual information, decision trees do not allow determination of the activation or repression relationship between the genes.

2.1.6 Gaussian Process

The Gaussian process (GP) is a distribution over continuous functions. Gaussian process methods fit a gene Y's expression Y_t as as a function of its parents $pa(Y)$' s expression levels $X^pa(Y)$, where the function is drawn from the Gaussian process distribution.

$$Y_t | X^{pa(Y)}_{t-1} \sim \mathcal{N}(Y_t | \mu(X^{pa(Y)}_{t-1}), K(X^{pa(Y)}_{t-1})) \tag{2.7}$$

As with decision trees, these seek to allow more flexible relationships between genes without having excessive parameters to fit. Gaussian processes, like dynamic Bayesian networks, perform a search over causal graphs. As a result, they still suffer from the same scalability issues, requiring users to limit the number of possible regulators for a given gene. Typically this is set to 2 or 3, resulting in $O(G^2)$ and $O(G^3)$ scaling respectively, where G is the number of genes in the dataset.

Two primary methods have been developed, GP4GRN, which fits the derivative and uses separate hyperparameters for each model [1], and CSI, which fits the raw expression level and shares hyperparameters across genes [37]. CSI is the best performing method on the DREAM4 100-gene challenge in both AUROC and AUPR, while GP4GRN has also been highly effective [38].

2.2 Comparison of Methods

Many studies have evaluated the effectiveness of these causal inference methods in simulated and real gene expression data. We focus on those involving vector autoregression both for concision and because our method is based on vector autoregression. The main finding is that VAR performs effectively on data of similar time interval and high dimensionality to our GR dataset, which has a 1-hour interval and thousands of genes (Chapter 3) [25, 57]. However, in other studies, VAR was inferior to the dynamic Bayesian network (DBN) and Gaussian process (GP) for shorter and low-dimensional time series [62, 37].

Lopes [25] assess vector autoregressions (VAR), dynamic Bayesian networks (DBN), and mutual information methods (MI) on three microarray datasets: a 2006-gene fly dataset with hour-long time intervals (primarily) across 22 hours [18], a E. Coli dataset with 10-50 minute time intervals across 5 hours [55] and a 1000-gene Yeast dataset with 5 minute time intervals across 2 hours [41]. The method accuracy was evaluated

by comparison with known interactions in a database. The authors found that the lag-1 vector autoregressive models, "VAR(1) + lars" and "simone", performed the best on the Fly dataset with AUPRC of over 0.39. This suggests that VAR methods are effective for data of the hour-long time intervals, which is the case for our GR data.

Yao also compare vector autoregressions (VAR), dynamic Bayesian networks (DBN) and mutual information methods (MI) on data simulated from a hierarchical gene network. This data had the number of genes, 1000, exceeded the number of timepoints, 20. They find that top two methods are their own developed prior-knowledge VAR and the lasso-penalized VAR. They also found that the DBN could not complete when working with data of that scale, and that the MI methods did not perform as well as VAR. Yao's simulation work supports our use of VAR in the GR data: first, the simulated data is similar to our setting, with the same high-dimensionality and short time series, and second, it is computationally tractable.

Hlavácková-Schindler compared vector autoregression, copula vector autoregression, and dynamic Bayesian networks, finding that vector autoregresssions outperform and run faster than dynamic Bayesian networks. They concluding that DBNs are more well-suited to modelling local dynamics than larger networks [16].

Two studies found that VAR did not perform as well as other approaches. Zou specifically compare DBN and VAR, finding that DBN outperforms VAR on a short, simulated timeseries of 5 genes as well as better performance on a true clock network. They recommend that VAR be used on longer time series while DBN on shorter ones [62]. Meanwhile, Penfold provide the broadest comparison of vector autoregressions (VAR), ordinary differential equations (ODE), dynamic Bayesian networks (DBN), and Gaussian process (GP). On the simulated DREAM4 100-gene network with 21 timepoints, they found that GP outperformed DBN, which outperformed VAR/ODE. For the in-vivo networks (a 5-gene yeast network and 7-gene clock network), Penfold

again find that GP outperformed DBN, which outperformed VAR/ODE; all performed better than random.

These latter two findings suggest that for small-sized networks, GP or DBN are more effective than VAR. The main drawback is that both GP and DBN can be quite computationally challenging and require sophisticated implementation. Thus, we chose to still focus on VAR for its simplicity, interpretability, and speed.

2.3 Challenges in High-Dimensional vector autoregression

Here, we review vector autoregression methods organized by the challenges faced by high-dimensional gene regulatory network inference: confounding, and local and global statistical significance.

Any strong pairwise relationship between genes could be the result of confounding, because there are 18,000 genes in the genome which are often strongly correlated [53]. To handle this issue, genes are often regressed simultaneously using a shrinkage penalty to find a unique solution. Lozano [26] and Shojaie [50] introduce the Lasso penalty to regularize the VAR fit. The key contribution of each methods is a procedure for choosing the optimal lag. Opgen-Rhein use a James-Stein shrinkage method to regularize coefficients [36], while Yao introduce a 2-step procedure based on prior knowledge and Ridge Regression [57]. Besides shrinkage methods, Tam did a two-step procedure where they first screen possible predictors for a gene by using pairwise regressions, and then use those predictors in a joint fit for the gene [53].

In our work, we use the Elastic Net penalty, which has received less focus than the Lasso or Ridge penalties. Elastic Net is an ideal intermediary between Lasso and Ridge. Elastic Net is more robust to correlated predictors than Lasso; this is critical because there are many highly correlated genes in our time series data which may

jointly be involved in regulation, and the Lasso penalty will make an unstable choice one of these predictors [63]. Elastic Net also provides a sparse, more interpretable fit, unlike the Ridge penalty which has most coefficients set to be nonzero.

The second challenge is declaring whether a single cause-effect relationship between genes X and Y, is statistically significant. This is done using the Granger Causality framework, where X causes Y when including the information from X significantly improves one's ability to predict Y [13]. Significance is classically assessed using an F-test of whether using X significantly reduces the error in predicted output Y [35, 57]. However, the F-test fails in the high-dimensional setting because the degrees of freedom becomes undefined (i.e. not positive). Furthermore, sparse estimators such as the Lasso or Elastic Net have non-Gaussian limiting distributions with a point mass at zero. Thus, they do not provide valid confidence regions or p-values [6].

One can address the challenges of high-dimensional significance testing using either analytical or permutation-based approaches. Analytical approaches adapt classical significance testing to the high-dimensional setting. For example, Opgen-Rhein develop a significance testing framework based on partial correlation coefficients, which facilitates small-sample testing and accounts for dependencies among estimated coefficients [36]. Buhlmann discuss two approaches developed in the statistics literature: multisample splitting and projection [6]. Multisample splitting involves splitting a sample into a training and validation set. In the training set, one uses the Lasso penalty to select the possible predictors (i.e. causes) for a given output (i.e. effect). This returns us to the low-dimensional setting where the number of predictors lies below the number of samples. In the validation set, one then fits this low-dimensional model and performs classical significance testing. Multiple sample splits are performed to get a p-value that is robust to the choice of split. Another approach is to use projection, a bias-corrected Lasso estimator, which has a Gaussian limiting

distribution [6]. We chose not to use analytical approaches because this still an active area of research and many assumptions must be made for these approaches to be consistent.

One can also use permutations to generate null hypotheses for significance testing. These have the appeal of being less model-dependent and more data-driven than the analytical approaches. Yao attempt to use a Monte Carlo simulation to test the null hypothesis of a zero coefficient [57]. However, their simulation is from a uniformly random sample that differs in multiple ways from the empirical distribution, including in mean and variance, which risks having excessively liberal rejections. Alternatively, Buhlmann suggest a stability selection method, in which the regression model is fit on multiple sub-samples (or bootstrap samples) and the frequency of a certain variable's appearance in the model is computed [6]. This proportion can then be linked to a False Positive error metric. This stability selection is especially elegant because it can be applied to any model. As long as data can be inputted and results can be outputted, stability selection can be performed.

In our work, we use both permutations and stability selection for our high-dimensional significance testing. We first develop an edgewise statistical null for declaring significant individual networks. This serves as a replacement to the undefined F-test used in the classical setting. Unlike Yao, we perform permutations directly on the original data and thus preserve properties of the empirical distribution such as the mean and variance. This prevents edges from being declared significant solely due to unnecessary differences between the original and null distributions. Then, we develop a global null distribution of possible bootstrap frequencies and use this to perform FDR thresholding of our networks. This is inspired by Buhlmann's stability selection; edge confidence is assessed by its frequency under small perturbations to the input data.

Chapter 3

Data

3.1 DREAM4 Network Inference Challenge

The DREAM4 100-gene Network Inference Challenge is the most comprehensive community benchmark of Gene Regulatory Network Inference methods, to our knowledge. At least 20 methods, not including our own, have been evaluated on this challenge [38, 20, 58, 19, 12]. We work on the data of the 100-gene network inference challenge from time series.

There are 5 datasets, each with its own simulated network of 100 genes. Each dataset consisted of 10 timeseries of 21 timepoints. The initial condition is set to the steady state measurement of a wild-type. For the first half of the timeseries, a "drug perturbation" is applied; this affects about 1/3 of genes. For the second half, the perturbation is removed and the system is allowed to relax back to the wild-type state. Network topologies were generated based on subnetworks from known transcriptional regulatory networks of *E. coli* and *S. cerevisiae* [29].

Dynamics of the networks were simulated using a detailed kinetic model of gene regulation. This included both independent and synergistic gene regulation. Moreover, both transcription and translation are modeled. Though protein concentrations

are calculated, only mRNA concentrations are provided as in the real-life situation of gene expression assays. Simulations use stochastic differential equations to model internal noise in network dynamics. In addition, measurement noise, using a similar model to a mix of normal and lognormal noise, is added to the expression datasets [29].

Evaluation is typically done by looking at the Area Under the Precision Recall curve (AUPR) or the Area Under the Receiver Operating Characteristic (AUROC). Thus, any network inference method that provides a ranking of possible edges can be evaluated on this dataset. Indeed, even methods that do not provide a method to threshold the network (e.g. to control the proportion of false positives), such as decision tree methods like dynGENIE3 [12] and Jump3 [19], can be evaluated [29].

3.2 Glucocorticoid System

3.2.1 Joint-unperturbed Gene Expression Data

We study gene expression data from the Glucocorticoid Receptor project; henceforth we will refer to this as the "GR" data. There was an "original" dataset of 4 replicates that stimulated by the glucocorticoid dexamethasone, and an "unperturbed" dataset of 3 replicates that allowed the system to relax after stimulation by dexamethasone. Each profiled gene expression across 12 timepoints.

We integrated these into a "joint-unperturbed"" dataset with 7 replicates. The idea was that any causal relationship between genes should still be preserved under different environments, as claimed in causal graph theory [39]. This also increased the sample size of our dataset from the original dataset's 48 replicate-timepoint pairs to the joint-unperturbed dataset's 84 replicate-timepoint pairs.

Original Gene Expression Data

The description below of the original GR Data is largely drawn from my junior thesis [27]. I include it in full detail here for completion.

The glucocorticoid receptor (GR) regulates the transcription of a variety genes controlling the metabolism and immune response [46]. It is activated via binding to glucocorticoids; the bound complex then enters the nucleus and activates or represses the transcription of a variety of genes, both on its own and as bound to other proteins [46, 51, 7, 11, 31]. The GR dataset seeks the comprehensive characterization of the genomic response to glucocorticoids through the measurement of changes in chromatin accessibility, epigenetic state, transcription factor binding, chromatin looping, and gene expression at time points across 12 hours of glucocorticoid, specifically dexamethasone, treatment [31]. Gene expression is profiled using RNA-Sequencing [31]. We extracted the temporal profiles of the genes from the GR expression data set across the 12 time points: $\{0, 0.5, 1-8, 10, 12\}$ hrs of Dex exposure.

We select the temporal profiles of those genes whose average expression across time were higher than 2 TPM and that passed the edgeR [47] criteria for differential expression. To measure average expression, we first averaged the genes expression value per timepoint, and then took the average of those timepoint averages. For differential expression, we used the same method as in [31]: for each timepoint, we tested each genes expression against its basal expression at an FDR threshold of 0:05, such that the resulting selected genes had expression different from the basal expression for at least one time point. These steps lead to a processed data set of 2767 differentially expressed genes. Finally, we added NR3C1, which encodes the GR transcription factor. NR3C1 was not found to be differentially expressed at the FDR threshold of 0.05, but was so at an FDR threshold of 0.2 [31]. In the end, we had a set of 2768 genes, which included 226 transcription factors. The resulting temporal profiles were further log transformed (base e) and corrected for surrogate variables

using SVAseq [22].

There were 4 replicates of the GR gene expression dataset across time. We split the data by replicates. All replicates besides replicate 1 had a measurement for each timepoint. Replicate 1 was missing timepoints 5 and 6 hrs, so we performed a linear imputation for these values in the log-transformed, surrogate-corrected space.

Unperturbed Gene Expression Data

The unperturbed gene expression dataset intends to measure the abatement of the glucocorticoid response; i.e. the return of the system to the basal state once drug exposure ends. While in the original dataset, gene expression is measured over the 12-hour course of continued dex exposure, in the unperturbed dataset, the system is exposed for 12-hours, and then the conditioned media is replaced and the drug is removed. Thus, gene expression is measured over an additional 12-hour timecourse after the drug removal.

There were 3 replicates of this dataset across time. As in the original data (Section 3.2.1), the unperturbed gene expression consists of the time points $\{0, 0.5, 1-8, 10, 12\}$ starting at the point of dex removal. As in the original data, the resulting temporal profiles were further log-transformed (base e) and corrected for surrogate variables using SVAseq [22]. The same set of 2768 genes and the same normalization scheme was used as in the original data.

3.2.2 Over-expression Data

In order to understand the role of various transcription factors in the GC response, each was separately over-expressed concurrently with drug exposure to generate a new dataset. By over-expressed, we mean that the expression level is maintained at a constant higher level than would be sustainable under biological conditions. Besides the over-expressed gene, all other experimental conditions such as drug exposure and

cell media as in the original gene expression data. Thus one can better isolate the causal effect of that specific gene.

For each of the transcription factors $CEBPB$, $CEBPD$, $FOSL2$, $FOXO1$, $FOXO3$, $KLF6$, $KLF9$, $KLF15$, $POU5F1$, and $TFCP2L1$, a separate over-expression dataset was created. This dataset had 3 replicates. Each replicate consisted of measurements at 5 points during the 12-hour time course: $\{0, 1, 4, 8, 12\}$ hours after the start of the experiment. As with the previous datasets, gene expression values were log-normalized (base e) and corrected for surrogate variables using SVAseq [22].

Log Fold-changes

To assess the effect of the over-expressed gene on the system, log-2 fold-changes of genes between the over-expressed dataset and the original data were computed as follows. Each dataset is subsetted to the timepoints $\{1, 4, 8, 12\}$ hours after the start of the experiment. For each timepoint T and each gene G, G's expression values (across multiple replicates) at T is averaged. Then the log-2 fold change between G in the over-expression data and G in the original data is computed for T. Finally, of the log-2 fold changes for each timepoint, the one that is greatest in absolute value is chosen. (for example, if G's log-2 fold change is 0.5 for time 1 and -10 for time 2, then G's final fold-change is set to be -10).

3.2.3 Literature-curated Network

We validated our network on a literature-curated Glucocorticoid Receptor Regulatory Network, in which each edges was an experimentally validated regulatory interaction. This was provided by the NDeX database [42, 40]. The name of the network was "Glucocorticoid receptor regulatory network". Multiple edge types were listed; we limited only to those of type "controls-expression-of."

Data was given in the form of protein product names and needed to be transformed

into gene names. The corresponding gene for each protein was found using Uniprot [9]. Thus, regulatory edges between genes were found. Certain proteins, like the $Cbp/p300$, consisted of two gene products, so these were split into two genes and the corresponding edges were drawn for both nodes. The final network consisted of 95 edges among 61 unique genes. There were 29 unique causal genes and 31 unique effect genes.

We find minimal intersections of the curated network's gene set with our data's genes (Section 3.2.1). Only 5 of the 29 causal genes were present in our data: FOS, JUN, $NR3C1$, $NR4A1$, $STAT1$. Only 6 of the 32 effect genes were present in our data: AFP, BAX, $CXCL8$, FGG, $SGK1$, and $VIPR1$.

By limiting the literature-curated network only to those genes that were present in our data, we reduced it from 95 to 7 edges: $JUN \rightarrow AFP$, $NR3C1 \rightarrow SGK1$, $NR3C1 \rightarrow CXCL8$, $NR3C1 \rightarrow AFP$, $NR3C1 \rightarrow FGG$, $NR3C1 \rightarrow BAX$, and $NR3C1 \rightarrow VIPR1$. This was a sobering limitation. 6 out of 7 edges had $NR3C1$ (i.e. the gene that encodes the glucocorticoid receptor) as the cause. However, $NR3C1$ was lowly differentially expressed, at an FDR of 0.2 (Section 3.2.1).

3.3 Gene Annotations

The description below of the gene annotations is largely drawn from my junior thesis, [27]. I include it in full detail here for completion.

In Results (Chapter 5), we perform a variety of network analyses based on gene annotations. This section describes those annotations.

We used four main classifications in our analysis of genes: Immune, Metabolic, and Transcription Factor. We now describe the classification method.

Immune genes were called using two primary sources. The first is the Gene Ontology annotation "Immune" (*GO:0002376*) [2]. To emphasize experimentally verified

annotations, we only used the evidence codes EXP, IDA, IGI, IMP, IPI, IC, TAS. The second is the Gene Ontology Consortiums curated, ranked list of immune-related genes based off of multiple databases and experimental evidence [8]. For the GO annotation, We selected all those genes with score greater than or equal to 7. This resulted in 616 immune genes overall, and 109 immune genes in our list of 2768 genes (differentially expressed + GR).

Metabolic genes were called using two primary sources. The first is the Gene Ontology annotation "carbohydrate metabolic process" *GO:0005975* [2]. We only used the evidence codes EXP, IDA, IGI, IMP, IPI, IC, TAS. The second is the Gene Set Enrichment Analysis curated list of metabolic-related genes [52]. We searched only among those with experimental evidence: the Canonical, KEGG, BIOCARTA, and Reactome pathways. We used the following 4 search queries: "gluconeogenesis OR (glucose AND metabolism) OR glycolysis", "lipid AND metabolism", "Diabetes", "Obesity". We chose these queries to ensure we covered genes implicated in both metabolic processes and disorders, which may be affected by GR. Combining these, we found 544 metabolic genes overall and 120 in our list of 2768 genes. Finally, 65 genes were both immune and metabolic overall, and 12 were both immune and metabolic in our geneset.

Transcription Factors were called using the Bioguo database of Human Transcription Factors [59]. There were 1463 factors overall and 226 present in our list of 2768 genes.

GR direct targets were called based on the binding data of GR [31]. These genes were found to be up-regulated at timepoints 0:5, 1, or 2 hours after initial treatment with dexamethasone, and had GR binding within 10 kb of the transcription start site. Up-regulation was called based on a differential expression test between a genes expression at one timepoint with the basal timepoint at FDR 0:01. This calling method included several positive controls such as DUSP1 [49] and PER1 [44]. The

method resulted in 111 genes. All 111 of these genes were included in our set of differentially expressed genes by definition.

GR-associated genes were called first using the Gene Ontology annotation "glucocorticoid receptor signaling pathway" *GO: 0042921* [2]. We only used the evidence codes EXP, IDA, IGI, IMP, IPI, IC, TAS. The second set was from the Gene Set Enrichment Analysis curated list of metabolic-related genes [52], searching the Cahnonical, KEGG, BIOCARTA, and Reactome pathways. We used the search query "glucocorticoid" and chose the result that said "PID_REG", which is curated by Nature and the National Cancer Institute.

Chapter 4

Methods

4.1 Causal Inference Framework: BETS

Our method builds upon my junior thesis work [27], so sections 4.1.2 through 4.1.6 were largely drawn from my junior thesis. The key innovations of our method are discussed in Sections 4.1.7 and 4.1.8.

Bootstrap Elastic net inference from Time Series (BETS) is a vector-autoregressive approach to causal inference from gene expression time series data. It is based on the principle of Granger Causality [13], in which a gene g is said to be causal for another gene g' if using information from gene g significantly improves our ability to predict gene g'.

BETS uses the elastic net penalty to handle the high dimensionality of the time series. It infers statistically significant networks from bootstrapped samples and then declares a global significant network based on based upon the frequency of individual edges among these networks; this is inspired by the stability selection procedure outlined in Section 2.3.

The algorithm is run as follows:

1. **Preprocess** the data (Section 4.1.1).

2. Choose the **Hyperparameters** by minimizing the cross-validation error (Section 4.1.4).

3. **Fit the model** on the original data and on the gene-specific null (Section 4.1.2).

4. Set a **local statistical significance** threshold for each edge and declare a significant network (Section 4.1.5, 4.1.6).

5. Do steps 3 through 4 for 1000 **bootstrapped samples** (Section 4.1.7).

6. Do steps 3 through 4 for 1000 bootstrapped samples on an **"umbrella" null** (Section 4.1.8).

7. **Declare a significant global network** based on an edge's frequency among the bootstrapped networks, controlling the global False Discovery Rate (Section 4.1.8).

4.1.1 Preprocessing

Out of two possible normalization schemes for the temporal profiles: zero-mean unstandardized and zero-mean unit-variance, we choose the zero-mean unstandardized version. Zero-mean unstandardized centers each gene temporal profile to have zero-mean across time. Zero-mean unit variance centers each gene temporal profile to have zero-mean, and then standardizes it to have unit variance. By gene temporal profile, we mean the gene's expression values across time for a single replicate. Because the variance of gene temporal profiles ranged from almost constant to drastic increases and decreases, **we chose to focus on the zero-mean unstandardized normalization scheme** because a unit-variance normalization would over-represent the weak causal effects of genes with lower variability.

4.1.2 Vector Autoregression Model

We used a vector autoregressive model (VAR) with lag $L \in \{1, 2\}$ to fit temporal gene expression profiles across multiple replicates of G genes over $t \in \{1, 2, \ldots, T\}$ time points. Let $\mathbf{g}- \in \{1, 2, \ldots g-1, g+1, \ldots, G\}$. Let $X_t^g = [X_{t,1}^g, X_{t,2}^g, \ldots, X_{t,R}^g]^T$ be the $R \times 1$ vector of gene expression levels of gene g across R replicates at time t. (for the DREAM data, $R = 10$, while for the GR data, $R = 7$). We modeled each gene g as

$$X_t^g = \sum_{l=1}^{L} m_l^g X_{t-l}^g + \sum_{l=1}^{L} \sum_{g' \in \mathbf{g}-} \beta_l^{g',g} X_{t-l}^{g'} + \mu\epsilon_t \tag{4.1}$$

where $\epsilon_t \sim \mathcal{N}(0, 1)$. In other words, the expression of each gene g is modelled as a linear function of its and other genes' L previous expression values, under independent Gaussian noise. In Equation 4.1, m_l^g represents the (scalar) effect size of gene g's l-th previous value, X_{t-l}^g, on its current value, X_t^g. $\beta_l^{g',g}$ represents the (scalar) effect size of the l-th previous value of gene $g' \neq g$, $X_{t-l}^{g'}$ on gene g's current value, X_g^t. μ is the the intercept term. One should note Equation 4.1 requires that $t > l$ for the l-th previous value, X_{t-l}^g , to exist.

One should note that the VAR assumes equally spaced timepoints. The time interval for the GR data ranges from 0.5 up to 2 hours, and is therefore technically in violation of this requirement. A counterargument is that the short intervals are concentrated at the beginning (hours $0, 0.5, 1$) where there is more likely to be activity, and the long intervals are at the end (hours $8, 10, 12$), where there is likely decreased activity. Thus despite the theoretical violation, treating the timepoints as equally spaced may not be entirely problematic [25].

To demonstrate how our model is fit in practice, we reformulate Equation 4.1 using matrix notation. Here, each row represents one timepoint per replicate. There are $T - L$ timepoints with $t > L$ and R replicates, so there are $R(T - L)$ samples, or

rows, in total. Let $N = R(T - L)$.

Let

$$\mathbf{X}_t^g = \begin{bmatrix} X_{L,1}^g \\ \vdots \\ X_{L,R}^g \\ X_{L+1,1}^g \\ \vdots \\ X_{L+1,R}^g \\ \vdots \\ \vdots \\ X_{T,R}^g \end{bmatrix} \tag{4.2}$$

$\mathbf{X}_t^g$ is a $N \times 1$ vector. We can similarly write $\mathbf{X}_{t-l}^g$ which is the same vector, but replacing each entry with its l-th previous value. Now combine the L lagged vectors of gene g, $[\mathbf{X}_{t-1}^g, \ldots, \mathbf{X}_{t-L}]$ into $\mathbb{X}_{t-l}^g$, a $N \times L$ matrix of the L lagged values of gene g. Finally, let $\mathbf{m}_l^g$ be a $L \times 1$ vector of the L lagged coefficients.

$$\begin{aligned} \mathbb{X}_{t-l}^g &= [\mathbf{X}_{t-1}^g \ldots \mathbf{X}_{t-L}^g] \\ \mathbf{m}_l &= \begin{bmatrix} m_1^g \\ \vdots \\ m_L^g \end{bmatrix} \end{aligned} \tag{4.3}$$

Next, let us formulate the component of Equation 4.1 involving the other genes g' in matrix notation. Let $\mathbb{X}_{t-l}^{\mathbf{g}^-}$ be a $N \times L(G-1)$ predictor matrix of the genes $g' \neq g$. Each column is of form $\mathbf{X}_{t-l}^{g'}$. Note the number of columns is $L(G-1)$, because there are $G-1$ genes g' and for each gene g', there are L lagged values: $X_{t-1}^{g'}, \ldots, X_{t-L}^{g'}$.

$$\mathbb{X}_{t-l}^{\mathbf{g}^-} = \begin{bmatrix} \mathbf{X}_{t-1}^1 & \ldots & \mathbf{X}_{t-1}^2 & \ldots & \ldots & \mathbf{X}_{t-L}^G \end{bmatrix} \tag{4.4}$$

Let β_l be a $L(G-1) \times 1$ vector of the causal coefficients $\beta_l^{g',g}$ where $g' \neq g$.

$$\beta_l = \begin{bmatrix} \beta_1^{1,g} \\ \vdots \\ \beta_L^{1,g} \\ \beta_1^{2,g} \\ \vdots \\ \vdots \\ \beta_L^{G,g} \end{bmatrix} \tag{4.5}$$

We then seek to fit the model:

$$\mathbf{X}_t^g = \mathbb{X}_{t-l}^g \mathbf{m}_l + \mathbb{X}_{t-l}^{\mathbf{g}-} \beta_l + \epsilon_t \tag{4.6}$$

where ϵ_t is a $N \times 1$ vector with each element $\epsilon_{t,n} \sim N(0,1)$.

To write in the most compact form, we can write

$$\mathbb{X}_{t-l}^{\mathbf{g}} = [\mathbb{X}_{t-l}^g \mathbb{X}_{t-l}^{\mathbf{g}-}], \quad \bar{\beta} = \begin{bmatrix} \mathbf{m}_l \\ \beta_l \end{bmatrix} \tag{4.7}$$

Note that $\mathbb{X}_{t-l}^{\mathbf{g}}$ is a $N \times LG$ matrix and $\bar{\beta}$ is a $LG \times 1$ vector.

Thus in final form we would fit:

$$\mathbf{X}_t^g = \mathbb{X}_{t-l}^{\mathbf{g}} \bar{\beta} + \epsilon_t \tag{4.8}$$

With these equations prepared, we are ready to describe the penalized fitting procedure.

4.1.3 Penalized Regression

The ordinary least squares estimator fits the causal coefficients as:

$$\hat{\beta} = \underset{\arg\min}{\bar{\beta} \in \mathbb{R}^{LG}} \|\mathbf{X}_t^g - \mathbb{X}_{t-l}^{\mathbf{g}} \bar{\beta}\|_2^2 \tag{4.9}$$

Here $\|\cdot\|_2$ represents the l_2-norm of a vector, i.e. the square root of the sum of the sum of the vector's squared coordinates. However, we are in the high-dimensional setting: the dimension, LG exceeds the sample size $N = R(T - L)$. For example, if $L = 1$, the dimension $LG = 2768$ whereas our sample size $N = R(T - L) = 44$. As a result, the ordinary least squares estimator is undefined. We must instead resort to the use of penalized approaches such as LASSO (Least Absolute Shrinkage and Selection Operator) [54], elastic net [63], and ridge regression [17]. These are designed for $\hat{\beta}$ to be sparse (only a few nonzero coefficients) and shrunk (reduced in magnitude).

We discuss the elastic net penalty, which is a more general case of the ridge and lasso penalties. The elastic net fits the following objective:

$$\hat{\beta} = \underset{\arg\min}{\bar{\beta} \in \mathbb{R}^{LG}} \|\mathbf{X}_t^g - \mathbb{X}_{t-l}^{\mathbf{g}} \bar{\beta}\|_2^2 + \lambda(\alpha\|\bar{\beta}\|_1 + (1-\alpha)\|\bar{\beta}\|_2^2) \tag{4.10}$$

Here $\|\cdot\|_1$ represents the l_1-norm and $\|\cdot\|_2$ represents the l_2-norm.

By setting $\alpha = 1$ in the above equation 4.10, we obtain the Lasso objective function. By setting $\alpha = 0$ in the above, we obtain the Ridge objective function.

For the Elastic Net, we used the following ranges of hyperparameter values: $\lambda \in \{10^{-4}, 10^{-3}, \ldots, 1\}, \alpha \in \{0.1, 0.3, \ldots, 0.9\}$. For Lasso, we used $\lambda \in \{10^{-5}, \ldots, 1\}$. For Ridge, when we used $\{10^{-5}, \ldots, 1\}$, we found that the the optimal value selected in some cases was the max 1 [27]. We thus expanded the range to $\{10^{-5}, \ldots, 10^{6}\}$ to ensure that we were not missing more optimal hyperparameters at larger values. At this point, the optimal λ was found to be 100 (Table 4 in [27]).

4.1.4 Hyperparameter Tuning

Hyperparameters were selected using leave-one-out cross-validation (LOOCV). The hyperparameter (or pair of hyperparameters, for elastic net) that minimizes the mean-squared error on the held-out datapoints is selected. More specifically, we first fix a hyperparameter (λ, α). Then, for a given gene g and row index i, extract the i-the row of $\mathbf{X}_t^g$ and $\mathbb{X}_{t-l}^{\mathbf{g}}$. Refer to this extracted validation set as $(\mathbf{X}_t^g)_i$ and $(\mathbb{X}_{t-l}^g)_i$. The remaining data is the training set, $(\mathbf{X}_t^g)_{-i}, (\mathbb{X}_{t-l}^{\mathbf{g}})_{-i}$.

First we fit our coefficient $\hat{\beta}_{\lambda,\alpha}^{g,i}$ over the training set.

$$\hat{\beta}_{\lambda,\alpha}^{g,i} = \underset{\bar{\beta} \in \mathbb{R}^{LG}}{\arg\min} \|(\mathbf{X}_t^g)_{-i} - (\mathbb{X}_{t-l}^{\mathbf{g}})_{-i}\bar{\beta}\|_2^2 + \lambda(\alpha\|\bar{\beta}\|_1 + (1-\alpha)\|\bar{\beta}\|_2^2) \tag{4.11}$$

We then compute the fit's prediction error on the validation set, $\|(\mathbf{X}_t^g)_{-i} - (\mathbb{X}_{t-l}^{\mathbf{g}})_{-i}\hat{\beta}_{\lambda,\alpha}^{g,i}\|_2^2)$. We repeat the fit $\hat{\beta}_{\lambda,\alpha}^{g,i}$ and error for every row index i of $\mathbf{X}_t^g$ and for every gene g. The mean held-out cross-validation error for (λ, α) is:

$$MSE(\lambda, \alpha) = \sum_{g=1}^{G} \sum_{i=1}^{R(T-L)} \frac{1}{GR(T-L)} \|(\mathbf{X}_t^g)_i - (\mathbb{X}_{t-l}^{\mathbf{g}})_i \hat{\beta}_{\lambda,\alpha}^{g,i}\|_2^2 \tag{4.12}$$

The (λ, α) which minimizes the error in Equation 4.12 is selected.

4.1.5 Edgewise Statistical Null

We develop a statistical null to declare individual edges statistically significant. Borrowing from the language of econometrics, a gene g is Granger-caused by a gene $g' \in \mathbf{g}-$ if using the past values of g' can improve our prediction of gene g, given the information from all remaining genes. In the language of vector autoregression, this means that for at least one lag l, $\beta_l^{g',g}$ is significantly different from 0 [13]. The null hypothesis, where the $\beta_l^{g',g}$ is equal to 0, is evaluated using a permutation test.

In previous work, we explored two possible choices of nulls: the "global" and "local" null [27]. The global null permutes every possible causal gene, i.e. all of $\mathbf{g}-$, while the local null only permutes the particular causal gene g'. In both cases, the model is fit again over the permuted dataset to generate a null distribution of coefficients, under the case where the causal time structure ought to be removed. **Based on our previous results, we choose to use the global null**, because the local null is difficult to reject due to having higher causal coefficients (Section 6.4.1 of [27]). We describe both for completion.

We first generated a single permuted dataset $\tilde{X}_t^G$. For each gene, we independently shuffled the expression values of each gene $g \in \{1, \ldots, G\}$ across time. This is done separately for distinct replicates.

For the global null, we wish to model the hypothesis of no causal relations, from any gene $g' \in \mathbf{g}-$, upon a given effect gene g. Thus, we use the unpermuted values of the effect gene X_t^g and the permuted values of all other causal genes $g' \in \mathbf{g}-$, as $\tilde{X}_t^{\mathbf{g}-}$. Permuting the effect gene X_t^g would allow us to test the significance of the gene's self-interaction, but we are only interested in testing significance of the causal relations of other genes on the given gene. Thus we do not permute the effect gene g.

Null causal coefficients $\tilde{\beta}^{\mathbf{g}-}$ are then fit as

$$X_t^g \sim \mathcal{N}(\sum_{l=1}^{L} m_l^g X_{t-l}^g + \sum_{l=1}^{L} \sum_{g' \in \mathbf{g}-} \beta_l^{g',g} \tilde{X}_{t-l}^{g'}, 1) \tag{4.13}$$

For the local null, we wish to model the case of no causal relation from gene g' upon gene g. Thus, we only use the permuted values of the causal gene g', $\tilde{X}_t^{g'}$, and use the unpermuted values of the effect gene g, X_t^g and of all the remaining genes $\mathbf{X}_t^{\mathbf{g}}$

The null causal coefficient $\tilde{\beta}^{g'}$ is then taken from its fit in:

$$X_t^g \sim \mathcal{N}(\sum_{l=1}^{L} m_l^g X_{t-l}^g + \sum_{l=1}^{L} \sum_{g' \neq g,g'} \beta_l^{g',g} X_{t-l}^{g'} + \sum_{l=1}^{L} \beta_l^{g',g} \tilde{X}_{t-l}^{g'}, 1) \tag{4.14}$$

4.1.6 Dataset-specific Statistical Significance

We develop a framework for declaring a significant network from a given dataset. This can be applied to any dataset that is fit, for example the original GR data or a bootstrapped version of it.

Similar to the null model, we consider two alternatives for controlling the False Discovery Rate: a global approach and a local one. For a fixed lag, the global FDR controls the rate of insignificant causal relations across the whole inferred causal network, while the local FDR controls for the causal relations conditioning on the specific effect gene. The local FDR may be more appropriate when there is a stringent threshold for one effect gene; i.e. the null coefficients for the effect gene. Under the global FDR, this would lead to a stringent threshold for all effect genes, while in the local FDR, it would only lead to a stringent threshold for that specific coefficient. **Based on our previous results, we use the local FDR**, because the global FDR is difficult to reject due to having higher causal coefficients (Section 6.4.1 of [27]). We describe both for completion.

Let $\beta_l^{\cdot,g}$ refer to the set of all lag-l causal coefficients for the effect gene g. Let $\beta_l^{\cdot,\cdot}$ refer to the set of all lag-l causal coefficients. Define $\tilde{\beta}_l^{\cdot,g}$ and $\tilde{\beta}_l^{\cdot,\cdot}$ analogously for the null coefficients.

We control the global FDR by fixing a lag $l \in \{1, \ldots, L\}$ and finding the threshold T_l such that

$$\frac{|\{|\tilde{\beta}_l^{\cdot,\cdot}| > T_l\}|}{|\{|\tilde{\beta}_l^{\cdot,\cdot}| > T_l\} + |\{|\beta_l^{\cdot,\cdot}| > T_l\}|} < 0.05 \tag{4.15}$$

For each gene pair (g', g), $g' \in \mathbf{g}-$, a causal link $g' \to g$ exists if for at least one of the lags $l \in \{1, \ldots, L\}$, $|\beta_l^{g',g}| > T_l$

We control the local FDR by fixing a lag $l \in \{1, \ldots, L\}$ and an effect gene g and finding the threshold T_l^g such that

$$\frac{|\{|\tilde{\beta}_l^{\cdot,g}| > T_l^g\}|}{|\{|\tilde{\beta}_l^{\cdot,g}| > T_l^g\} + |\{|\beta_l^{\cdot,g}| > T_l^g\}|} < 0.05 \tag{4.16}$$

¡ 0:05 (15) For each gene pair (g', g), $g' \in \mathbf{g}-$, a causal link $g' \to g$ exists if for at least one of the lags $l \in \{1, \ldots, L\}$, $|\beta_l^{g',g}| > T_l^g$ The only difference from the global FDR is that there is a threshold T_l^g specific to the effect gene g.

4.1.7 Bootstrapped Networks

Our motivation for fitting bootstrapped data was to understand the stability of the inferred network. How would the network change if it were instead inferred from a bootstrapped sample of the original data? Which edges would be preserved? As acknowledged in [32], such questions are natural to ask, related to stability selection. The authors apply stability selection to the problem of graph estimation (including regression, graphical modeling, or cluster analysis), which on subsamples/bootstrap samples and computes the frequency by which an edge appears. They prove that it provides finite sample control for the Family-Wise Error Rate. However, we are more interested in controlling the False Discovery Rate.

Refer to Figure 4.1. The key idea is to repeat the whole procedure described in Sections 4.1.5 and 4.1.6. $B = 1000$ bootstrap samples are drawn from the data used for the vector autoregression, i.e. the rows of $\mathbb{X}_{t-l}^{\mathbf{g}}$ are sampled from replacement. (in Figure 4.1, they are jointly represented as X_t). Recall each row $\mathbb{X}_{t-l}^{\mathbf{g}}$ of represents a distinct sample: a distinct timepoint $\times$ replicate pair, for example timepoint 5 of replicate 3. Thus, in total there are $(T - L)R$ rows.

For each bootstrap sample, the procedure described in Sections 4.1.5 and 4.1.6 is performed: the original coefficients are fit. A set of null coefficients are also fit from the edgewise permutation. These are used to declare a significant network. This procedure is repeated for each bootstrapped sample. At the end, **each edge's boot-**

strap frequency, i.e. frequency among the significant bootstrap networks, is computed.

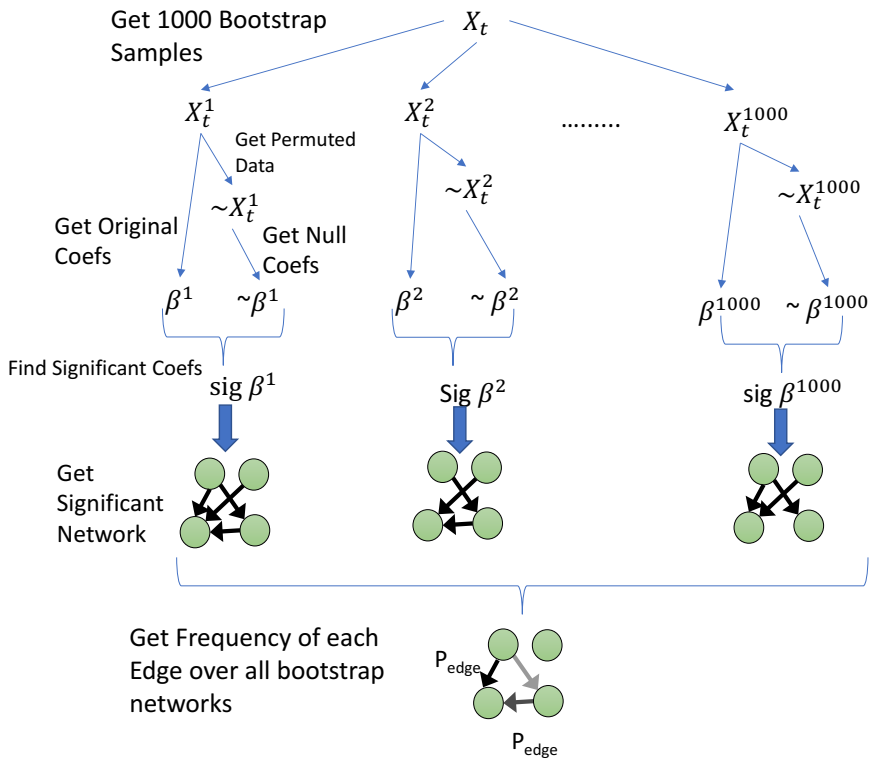

Figure 4.1: **Illustration of Bootstrap Frequency Procedure for assessing the robustness of edges.** The regression procedure is performed for $B = 1000$ bootstrap samples and significant networks are declared. Then, one counts the frequency of each edge's appearance across the significant networks.

4.1.8 Global Statistical Significance

From the previous sections, we have produced a set of bootstrap frequencies $\pi_{g',g}$ for each edge $g' \to g$. To determine the appropriate cutoff for the network, we generate a null distribution of bootstrap frequencies: first, we generate a second permuted dataset in which each gene's expression levels are independently randomized across time. Then, we run the steps of sections 4.1.2 through 4.1.7 on this permuted dataset to get the null bootstrap frequency of each edge, $\tilde{\pi}_{g',g}$.

Finally, we control the global FDR at 0.2 by finding the threshold T_b such that

$$\frac{|\tilde{\pi}_{g',g} > T_b|}{|\tilde{\pi}_{g',g}| + |\pi_{g',g}|} < 0.2 \tag{4.17}$$

We illustrate the corresponding distributions and possible thresholds T_b (Figure 4.2).

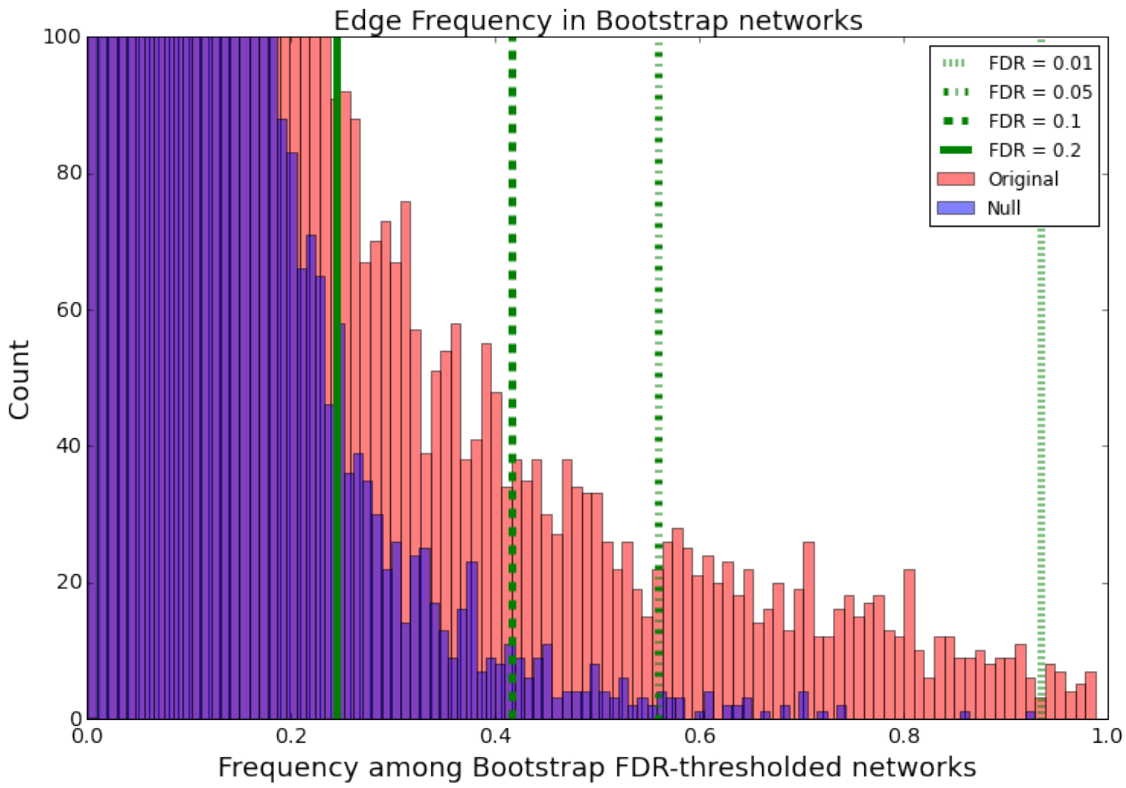

Figure 4.2: **FDR thresholds for Bootstrap Frequency Procedure over the original and null distributions.** For a given FDR, the corresponding green threshold T_b is found in the bootstrap frequency space. Edges in the original (red) distribution with bootstrap frequency above T_b are declared significant.

4.1.9 Implementation

We implemented BETS in Python2, using the libraries numpy, scipy, pandas, matplotlib, and sklearn. BETS implements the Ordinary Least Squares, Ridge, Lasso,

and Elastic Net fitting, while provides functionality for new versions to be introduced. All control, including run settings, data locations, and computational resources, are set in a Bash settings file. Running it consists of running a sequence of shell commands. Currently, data normalization and randomization (for the two statistical nulls) must be performed prior to running BETS; we hope to do this automatically in the future.

BETS is designed to handle large datasets and to take advantage of parallelization capabilities; our runs heavily used Princeton's della cluster. In each case of parallelization, individual scripts are created, which can be run independently in parallel. Computation is parallelized across the number of regressions (in all cases), across hyperparameters (in the case of cross-validation), and across bootstrap samples (in the case of bootstrap runs). Individual output files are then integrated in an online manner, in which files are loaded, integrated, and then deleted to prevent memory overflows. Because of the size of the coefficient matrices generated from each of the bootstrap samples, BETS uses large amounts of memory: for 1000 bootstrap samples in the 100-gene DREAM data, it used about 16 GB. To handle this, once aggregation is performed, individual bootstrap output files can be deleted. Scripts for doing all of these are generated automatically; one simply needs to run them in sequence.

Timing statistics are recorded both in real and CPU time, and are summarized at the end as well.

BETS will be available for download on Github. The results from all analyses are currently available upon request, and will be posted online in the spirit of scientific reproducibility.

4.2 Other Compared Methods

We also ran Jump3 [19] and CSI [37], two methods that outperformed BETS in AUROC and AUPR. We only discovered the two other top-performing and faster methods, dynGENIE3 and tl-CLR in a paper published just this year and so were not able to implement and compare them [12]. For Jump3 and CSI, we used the same framework of having all control set in a Bash settings file, and automatically generating scripts to run in parallel and integrate the results. Jump3 was in Matlab. CSI was in Python3 though the default version was in Matlab; Dr. Penfold graciously provided the Python3 version and relevant scripts upon request.

Certain parameters required author correspondence to be set because they were not reported in the papers. Jump3 requires one to set a "systematic noise" term describing intrinsic noise to the system and an "observational noise" term describing error from the measurement of gene expression. From correspondence with Dr. Huynh-Thu, on the DREAM4 data, we set the parameters in accordance with her previous run: the systematic noise was set at $1e-4$ and the observation noise at 0.01 times the value of the gene's expression.

In the Matlab version of CSI, which I was working with until Dr. Penfold provided the Python3 version, CSI requires one to set the hyperparameters of the gamma distributions used as Bayesian priors on the length scale and the process variance of the Gaussian process (recall that CSI models the nonlinear relationship between genes as a Gaussian process). Unfortunately, I was unable to find the parameters used in the run documented until I spoke with Dr. Penfold. The unavailability of these parameters online posed a risk for reproducibility. They turned out to be the default values in the Matlab version, $a = 10$ and $b = 0.1$, but it was important to confirm, to ensure that any discrepancies in the re-run would not be confounded by different parameter settings.

Finally, we ran the three Mutual Information methods ARACNE [30], MRNET

[33], and CLR [10] directly in R using the minet library [34]. No parallelization was needed as each methods ran in less than 5 seconds on the DREAM4 100-gene data.

The results from all analyses are available upon request, and will be posted online in the spirit of scientific reproducibility.

Chapter 5

Results

5.1 DREAM

5.1.1 Vector Autoregression

Before developing BETS, we had simply implemented the elastic net regression and significance thresholding of Sections 4.1.1 to 4.1.6 [27]. To assess this performance on DREAM, we ran the regression under the zero-mean unstandardized and the zero-mean unit-variance normalizations, with lags 1 and 2, under the elastic net penalty.

We find that the results of each run version are approximately the same, with both the AUROC and the AUPR all being within standard deviation of each other (Table 5.1.1). The zero-mean-unnormalized and lag 1 had the highest AUROC of 0.674, while the zero-mean unit-variance and lag 2 had the highest AUPR of 0.12. The results from each method was poor-performing compared with the rest of the methods assayed in the literature (ranking at about 7th in AUROC among 16 methods) [38, 19, 58, 12]. Although zero-mean-unnormalized and lag 2 did not have the best performance, we chose to continue to use it because it was almost equivalent with the others, because it could capture longer-term and more nonlinear interactions with the lag 2 (as we found in our Simulation Study in [27]), and because, as discussed in Section 4.1.1, we

did not want to amplify the representation of low-variance timeseries.

Run	Normalization	Lag	AUROC (average)	AUROC (stdev)	AUPR (average)	AUPR (stdev)
0mean1var-enet-1	0mean1var	1	0.646	0.03	0.118	0.03
0mean1var-enet-2	0mean1var	2	0.652	0.03	**0.12**	0.02
0mean-enet-1	0mean	1	**0.674**	0.05	0.112	0.03
0mean-enet-2	0mean	2	0.662	0.05	0.098	0.02

Table 5.1: **Results of Elastic Net Regression on DREAM4 100-gene Networks.** Under the Normalization column, "0mean1var" denotes normalization to zero-mean and unit-variance. "0mean" only denotes normalization to zero-mean.

To further improve the method, we tested the bootstrap stability selection procedure (Section 4.1.7, inspired by [6]). We used $B = 1000$ Bootstrapped samples.

We find that for every combination we tested: the elastic net penalty with lags 1 and 2, and the ridge and lasso penalties with lag 2, the Bootstrap version of the method outperformed the original version (Table 5.1.1). AUROC improved from 0.018 to 0.04 points while AUPR improved from 0.016 to 0.03. In each case the improvement is on the order of one standard deviation. We compare the results to the community benchmark methods in Section 5.1.2.

Run	Lag	Penalty	Coefficient AUROC	Bootstrap AUROC	Coefficient AUPR	Bootstrap AUPR
0mean-enet-1	1	Elastic Net	0.674 (0.05)	0.686 (0.05)	0.112 (0.03)	0.14 (0.03)
0mean-enet-2	2	Elastic Net	0.662 (0.05)	0.688 (0.06)	0.098 (0.02)	0.128 (0.02)
0mean-lasso-2	2	Lasso	0.652 (0.05)	0.692 (0.06)	0.14 (0.04)	0.162 (0.05)
0mean-ridge-2	2	Ridge	0.642 (0.04)	0.66 (0.05)	0.08 (0.03)	0.096 (0.03)

Table 5.2: **Improvement on DREAM4 100-gene Network Inference from Bootstrap.** For each AUROC or AUPR column, the average is the listed value and the standard deviation is listed in parentheses. "Coefficient" denotes the result when ranking edges by their fitted coefficient, as in the original method. "Bootstrap" denotes the results when ranking edges by the frequency by which they appear in the bootstrap networks.

Finally, we tested the sensitivity of our method to the number of bootstrap samples used. This is important for those with constraints on time and memory. We compared between 100 and 1000 samples (Table 5.1.1). Our results suggest that 100 would offer comparable performance to the 1000 case, given similar AUROC values (0.68 and 0.688, less than 2 standard deviations away from each other), and similar AUPR values (0.124 and 0.128, 2 standard deviations within each other). At the same time, the 100-sample version uses about 10 times less memory: 1.6 GB vs 15.6 GB. It also uses about 3 times less time: 1.6 hr vs 4.8 hr. Note that the time does not decrease by a factor of 10 because in the low bootstrap number regime, the time is dominated by the search over the hyperparameter space during cross-validation, and we evaluate 25 possible hyperparameter settings for the elastic net.

Algorithm	Bootstrap Samples	AUROC	AUPR	Time (hr)	Memory (GB)
0mean-enet-2-boot-100	100	0.68 (0.05)	0.124 (0.02)	1.6	1.6
0mean-enet-2-boot-1000	1000	0.688 (0.06)	0.128 (0.02)	4.8	15.6

Table 5.3: **Comparison of Bootstrap.** DREAM results reported for both 100 and 1000 bootstrap samples. All values in the columns are averages and the parenthetical values as standard deviations across the 5 DREAM4 Networks.

5.1.2 Community Benchmark

We next performed a literature review, finding 20 other methods that have been tested on the DREAM data [38, 19, 58, 12]. Of them, we find that 15 have been tested in AUROC.

BETS performs 3rd out of 17 methods in AUROC (Figure 5.2) and 6th out of 22 methods in AUPR Figure 5.1). BETS is the top performer of all vector autoregression methods, and Enet (our original method with ranking by coefficient [27]) is a close second. We also find that our VAR methods, BETS and Enet, have similar performance to the DBN methods in AUPR and outperform most of them in AUROC. This suggests that VAR is still an effective tool for regulatory network inference, contrary to the findings in [38]. Meanwhile, dynamic Bayesian networks and ordinary differential equations appear less effective.

All methods performed better than random, which performed at 0.5 AUROC and 0.002 AUPR [38]. The top-performing methods in AUPR were based on Gaussian processes (CSId, GP4GRN), decision trees (dynGENIE3, Jump3), mutual information (tl-CLR), and vector autoregression (BETS). Among these, dynGENIE3, tl-CLR, and BETS are the fastest, taking on the order of minutes to hours whereas the others take tens of hours (Table 5.1.2) [12].

We also note that CLR and MRNET had strong performance on DREAM4 in [58] by using an adapted procedure in which a gene's previous expression was corrected. However, we were unable to reproduce those favorable results after running their same adapted procedure, and produced equivalent results.

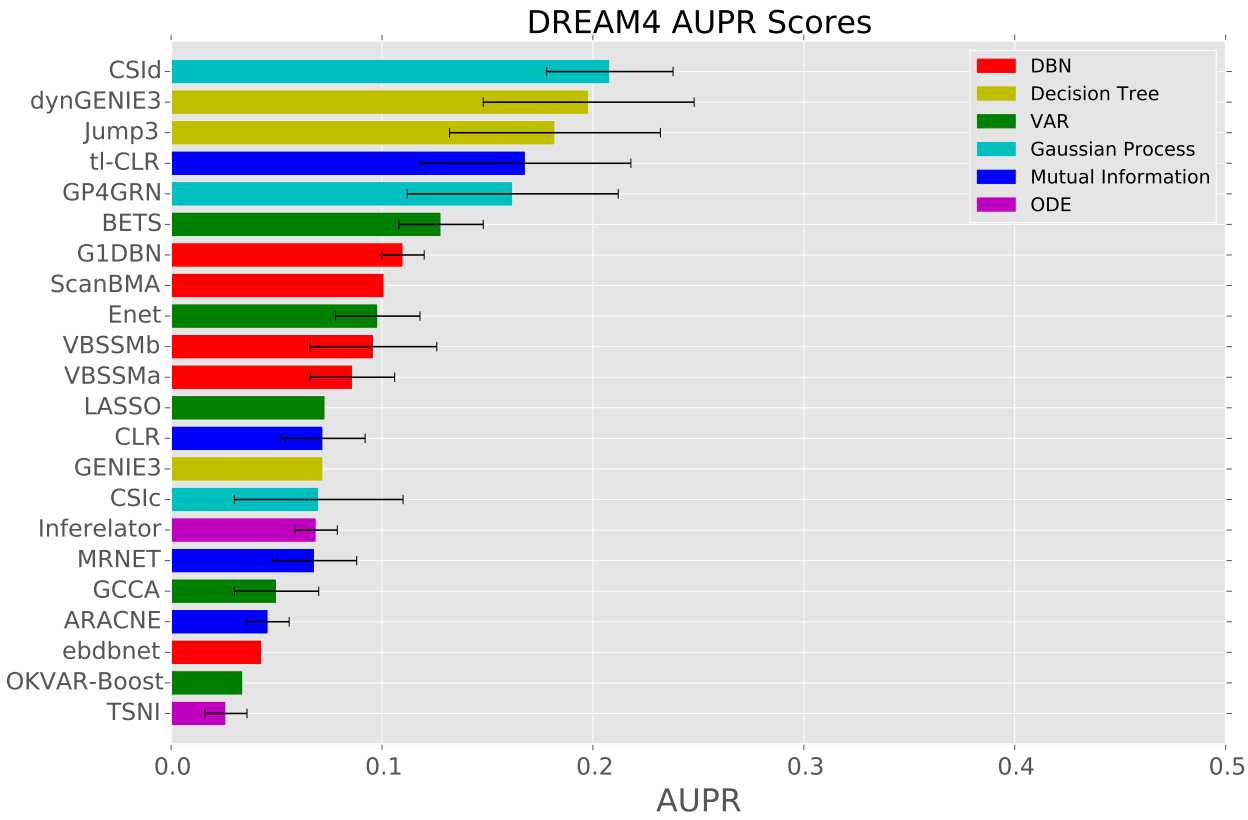

Figure 5.1: **DREAM4 AUPR Results.** Average values across the 5 DREAM networks are displayed. Bars reach one standard deviation away from the average. "DBN" denotes dynamic Bayesian network, "VAR" denotes vector autoregression, and "ODE" denotes ordinary differential equation. Algorithms that were run in-house were ARACNE, BETS, CLR, CSId, Enet, Jump3 and MRNET. These were consistent with reported literature values. Values for CSIc, G1DBN, GCCA, GP4GRN, TSNI, VBSSMa and VBSSMb were taken from [38]. Values for ebdnet, LASSO and ScanBMA, were taken from [58]. Values for dynGENIE3, GENIE3, OKVAR-Boost and tl-CLR were taken from [12]. Value for Inferelator, Jump3 and were taken from [19].

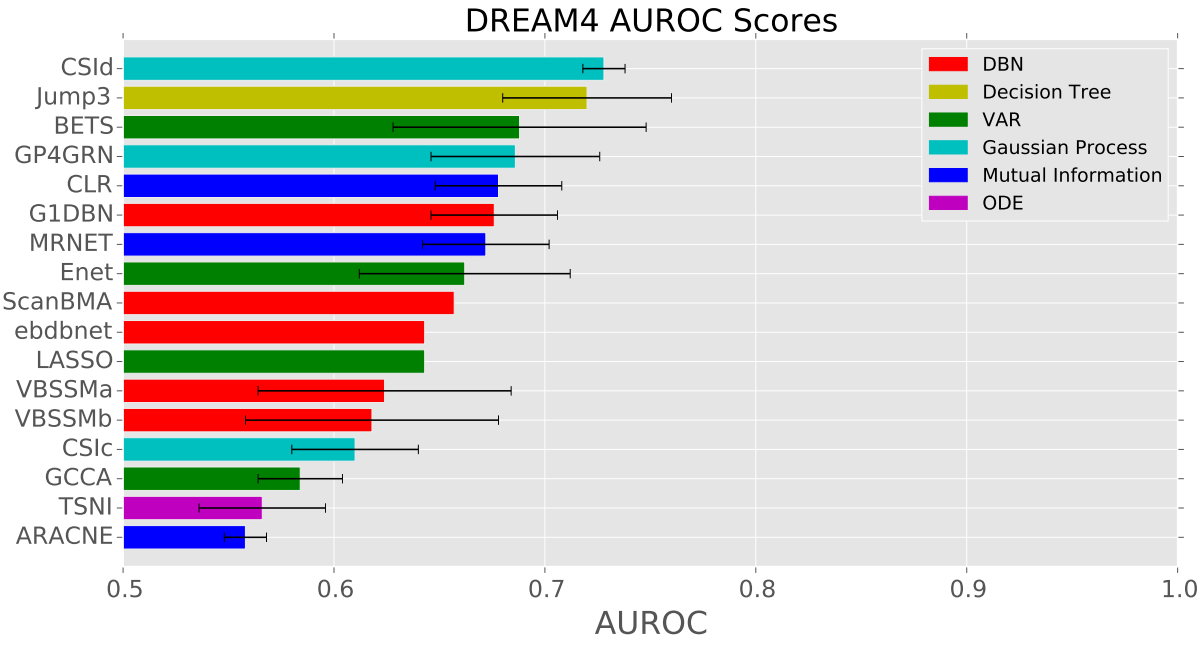

Figure 5.2: **DREAM4 AUROC Results.** Average values across the 5 DREAM networks are displayed. Bars reach one standard deviation away from the average. "DBN" denotes dynamic Bayesian network, "VAR" denotes vector autoregression, and "ODE" denotes ordinary differential equation. Algorithms that were run in-house were ARACNE, BETS, CLR, CSId, Enet, Jump3 and MRNET. Values for CSIc, G1DBN, GCCA, GP4GRN, TSNI, VBSSMa and VBSSMb were taken from [38]. Values for ebdnet, LASSO and ScanBMA, were taken from [58]. Value for Inferelator and Jump3 were taken from [19].

Finally, we benchmarked the timing results of the methods that we ran in-house: ARACNE, BETS, CLR, CSId, Enet, Jump3, and MRNET (Table 5.1.2). The most effective methods, CSI and Jump3, also took at least 9 hours to run on the small 100-gene dataset. Meanwhile, the mutual information methods are extremely fast. Among these methods, BETS strikes a good middle ground at being effective, at an AUROC of 0.688, while being reasonably fast, taking 5 hrs in CPU time.

Method	AUPR	AUROC	Time (s)	Time (hr)
CSId	0.208	0.728	33268	9
Jump3	0.182	0.72	162397	45
BETS	**0.128**	**0.688**	**17383**	**5**
Enet	**0.098**	**0.662**	**4149**	**1**
CLR	0.072	0.678	0.032	0
MRNET	0.068	0.672	0.038	0
ARACNE	0.046	0.589	0.036	0

Table 5.4: **Timing Results of In-House Algorithms.** Methods are listed in decreasing order of their AUPR values. BETS and Enet are bolded to indicate that they are our own developed methods, based on vector autoregression. CSId is a Gaussian process method. Jump3 is a decision tree method. CLR, MRNET, and ARACNE are mutual information methods.

5.2 Glucocorticoid System

We performed two runs of BETS on the joint-unperturbed data 3.2.1. **We will refer to the lag-1 version as BETS-1 and the lag-2 version as BETS-2.** These runs were performed on the zero-mean centered data, using the edgewise null (Section 4.1.5) and the local FDR control on each bootstrap networks (Section 4.1.6), and then controlling the global network FDR based on the bootstrap frequency at an FDR of 0.2 (Section 4.1.8).

Parallel computation was a key enabler of this study. BETS-1 took 13 days to complete after parallelizing over 53279 scripts; it took 168 days in CPU time. BETS-2 took 6 days to complete after parallelizing over 55113 scripts; it took 292 days in CPU time. Note that the computation time decreased only two orders of magnitude instead of the four to five we would expect from the number of scripts. The main bottleneck for parallelization was simply waiting for the jobs to be scheduled. We attempted to run Jump3 and CSI on our data; Jump3 was not parallelized and so could not complete, while CSI had not terminated after 7 days when we last checked.

5.2.1 Network Analysis

For BETS-1 and BETS-2, we summarize network statistics (Table 5.2.1), degree distributions (Figure 5.3) , enrichment (as odd ratios) for certain kinds of annotated genes and edges (Figures 5.4, 5.5), mutual concordance, and transcription factor relationships to immunity and metabolism.

Network summariesare listed in Table 5.2.1. As expected, there is a 1-2 order of magnitude lower number of causal genes than effect genes. This concords with our understanding of gene regulatory networks as initiated by a few genes whose effects cascade [15, 57]. Moreover, at the same FDR of 0.2, BETS-2 has an order of magnitude greater size than BETS-2, at 31945 edges compared with 2098 edges.

This is due to BETS-2 being able to capture lag-2 effects in addition to lag-1effects.

Network	Lag	Hyperparameter	Causal Genes	Effect Genes	Genes	% All Genes	Edges	% All Edges
BETS-1	1	(0.1, 0.1)	67	1548	1580	57	2098	0.027
BETS-2	2	(0.01, 0.1)	466	2768	2768	100	31945	0.42

Table 5.5: **Results of BETS run on the joint-unperturbed GGR data.** Two runs, at lags 1 and 2, were performed, with the elastic penalty. This used an FDR threshold of 0.2. "All Genes" denotes the 2768 genes that were inputted. "All Edges" denotes the 7659056 possible causal edges among those genes.

The BETS-1 causal network contains 2098 edges and is relatively sparse at 0.027% of possible edges. Only 67 genes are causal for 1548 effect genes, so the vast majority of genes have 0 Out-degree and only an in-degree of $1-2$ (Figure 5.3). The BETS-1 Network has enrichment for transcription factors among causal genes (odds ratio: 2.8, Fisher's exact test $p = 0.002$) immune genes among causal genes (odds ratio: 4.0, Fisher's exact test $p = 0.001$), and metabolic genes among causal genes (odds ratio: 4.7 , Fisher's exact test $p = 0.0001$) (Figure 5.4) It has enrichment for edges with transcription factors as causes (odds ratio: 2.6, Fisher's exact test $p = 7.1e-55$) and metabolic genes as causes (odds ratio: 14, Fisher's exact test $p = 1e-128$) (Figure 5.5). Note $p = 1e-128$ indicates that the p-value produced by Scipy's Fisher exact test, , so we used the smallest positive floating number on our computer.

The BETS-2 causal network contains 31945 edges and is less sparse than the BETS-1 network, but still sparse at 0.42% of possible edges. 466 genes are causal for 2768 effect genes: all genes have an inward edge. , so the vast majority of genes have 0 Out-degree and only an in-degree of $5-20$ (Figure 5.3). The BETS-2 Network has enrichment for transcription factors among causal genes (odds ratio: 2, Fisher's exact test $p = 2.6e-05$) and immune genes among causal genes (odds ratio: 2.9, Fisher's exact test $p = 1.2e-06$) (Figure 5.4). It has enrichment for edges with immune genes as causes (odds ratio: 2.7, Fisher's exact test $p = 1e-128$) and metabolic genes as causes (odds ratio: 3.0, Fisher's exact test $p = 1e-128$) (Figure 5.5). Note again that $p = 1e-128$ indicates that the p-value returned by Scipy's Fisher exact test was 0.0, so we used the smallest positive floating number on our computer.

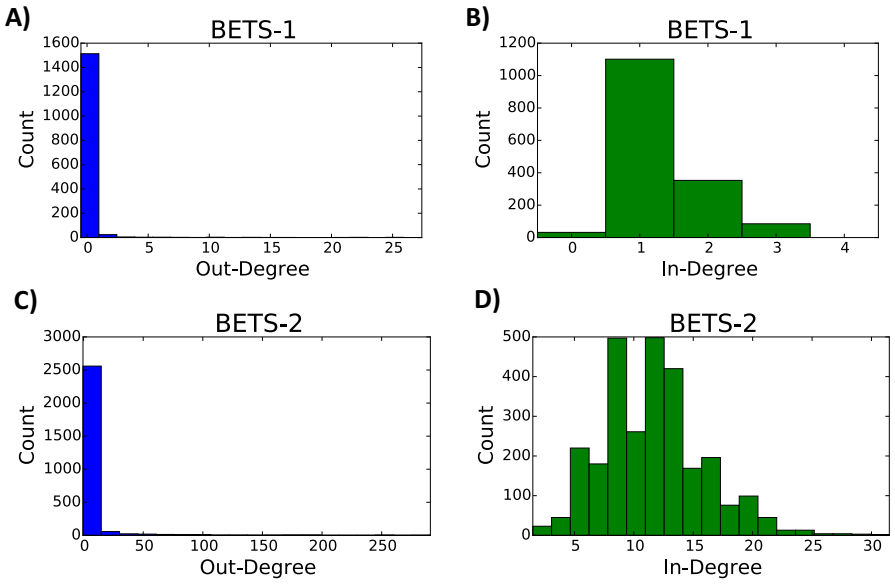

Figure 5.3: **Out- and In-Degree Distributions of genes in the 2 BETS-inferred Networks.** Out-degree distributions only display up to the 99th percentile because there are a few very high out-degree genes that would render the graph uninformative. Notice that there are higher in-degrees and out-degrees in the BETS-2 network, which has almost 15 times the number of edges as the BETS-1 network.

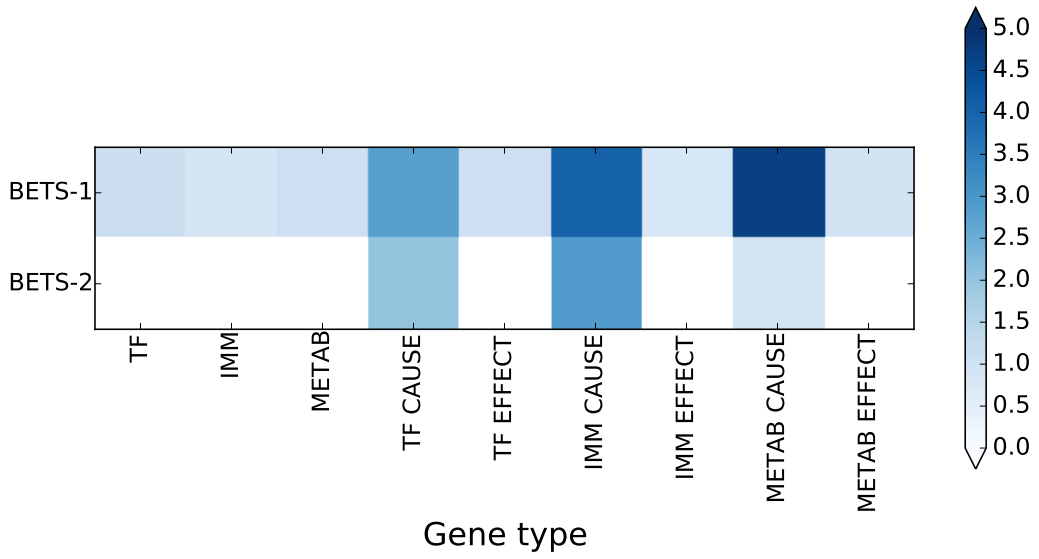

Figure 5.4: **Odds ratio enrichment for Gene Types in the 2 BETS-inferred Networks.** "TF", "IMM", "METAB", "ANY" refer to transcription factor, immune, metabolic, and any gene, respectively. A baseline odds ratio is 1.0; larger (darker) indicates enrichment. Note enrichment of TF, immune, and metabolic causal genes in BETS-1, and enrichment of TF and Immune causal genes in BETS-2.

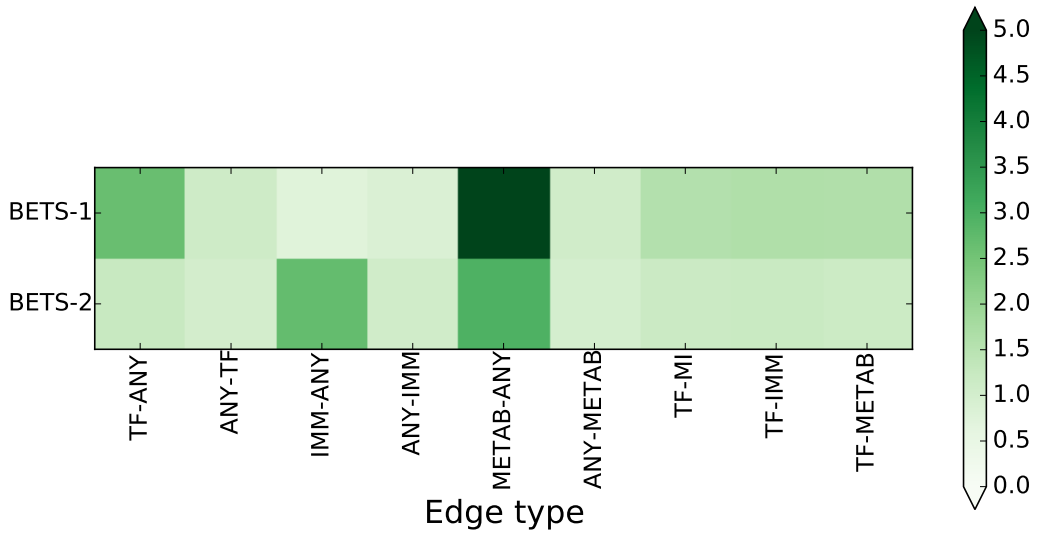

Figure 5.5: **Odds ratio enrichment for Edge Types in the 2 BETS-inferred Networks.** "TF", "IMM", "METAB", "ANY" refer to transcription factor, immune, metabolic, and any gene, respectively. A baseline odds ratio is 1.0; larger (darker) indicates enrichment. Note enrichment of TF and Metabolic-causal genes in BETS-1, and enrichment of Immune and Metabolic-causal genes in BETS-2.

There is high concordance between the BETS-1 and BETS-2 networks. There are 1286 shared edges between the BETS-1 network (2098 edges total) and the BETS-2 network (31945 edges total). The odds ratio of shared edges is 394, which is statistically significant at $p = 1e - 128$ for the Fisher's exact test (Scipy returned 0.0 for the p-value).

We are especially interested in the Transcription Factors that may be mediating the Metabolic and Immune effects. We chose to rank transcription factors by the proportion of their effects that were immune or metabolism related (Figure 5.6; Tables A, A).We see that in both the BETS-1 and BETS-2 network, $TFCP2L1$ and $ATF3$ have more metabolic effects than immune effects, while $TSC22D3$ is interemediate between the two, and $NR1D2$ has more immune effects than metabolic effects. There

are some interesting correspondences: in BETS-1, FOS an immune-gene has 0.75 of its immune-metabolic effects as immune (Table A). In BETS-2, $ATF3$ is a metabolic gene and has 0.56 of its immune-metabolic effects as metabolic.

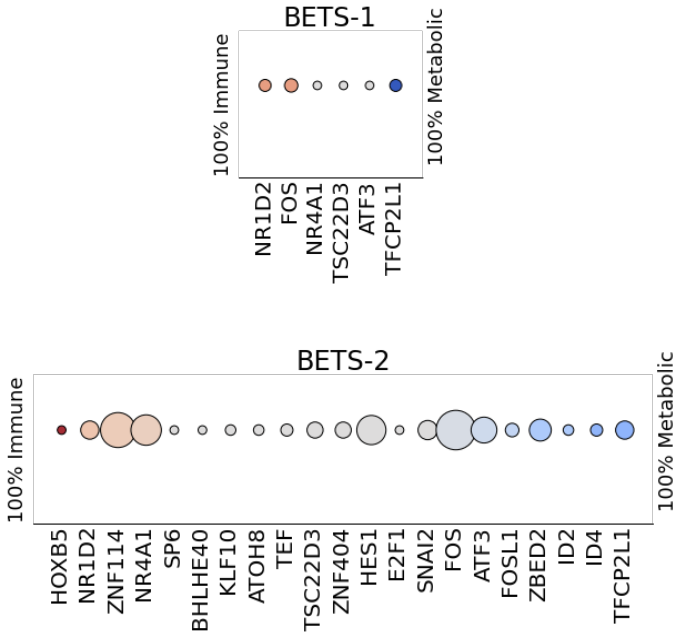

Figure 5.6: **Transcription Factors sorted by their immune and metabolic effects**. Transcription factors from each network with effects that were either metabolic or immune were taken. More red indicates a greater proportion of immune effects than metabolic effects; more blue indicates the opposite. The size of the dot indicates the number of the gene's effects that were either metabolic or immune.

5.2.2 Analysis of Bootstrap Procedure

We investigated the relationship of the coefficients estimated using the bootstrap coefficients with the original procedure (i.e. fitting simply elastic net). There were 276284 nonzero edges in the unthresholded network from the original fit, and 13095 nonzero edges after FDR-thresholding in the original fit. There were 2371217 nonzero edges in at least one of the unthresholded bootstrap networks, and 540711 nonzero

edges after in at least one of the FDR-thresholded bootstrap networks. Merging the edges that were nonzero in at least one network (either from the original fit or the bootstrap fit), we end up with 276282 edges, which is about 3.6% of all possible edges in the network. We next compare their coefficients in the original fit and bootstrap fits. Throughout this section, "significant edge" will refer to significance in the original fit, not the bootstrap fit.

We find a high correlation between the original coefficient (from unthresholded fit on the original dataset) and the mean bootstrap coefficient (coefficient averaged across all the **unthresholded** bootstrap fits). In each case, the mean bootstrap coefficient is a shrunk version of the original coefficient (Figure 5.7). Among the significant coefficients (Figure 5.7, middle column), the regression gives a fit of $y = 0.57x$ for Lag 1 ($r^2 = 0.93$) and $y = 0.54x$ for Lag 2 ($r^2 = 0.92$). Among the insignificant coefficients (Figure 5.7, right column), the regression gives a fit of $y = 0.50x$ for Lag 1 ($r^2 = 0.82$) and $y = 0.67x$ for Lag 2 ($r^2 = 0.82$).

When we consider the union of the coefficents, in which an edge's coefficient is set to be whichever of the lag 1 and lag 2 coefficients has a larger absolute value, we find slightly decreased correlation ($r^2 = 0.92$ for significant edges and $r^2 = 0.79$ for insignificant edges). We also find an interesting pattern in which several coefficients close to 0 have opposite signs between the original coefficient and the mean bootstrap coefficient– this is a diagonal line in the 2nd and 4th quadrants (Figure 5.7, top row). These may indicate unstable coefficients in the original fit, as bootstrap resampling causes the coefficient to flip; indeed, there appear to be more flips among the insignificant coefficients than the significant ones. In particular, they may be unstable due to the method of unionization: it may be the case that the original maximal coefficient was a certain lag, but the other lag had a higher (in absolute value) mean coefficient among the bootstrap coefficients. This would explain why this flip does not appear for the individual lag networks (Figure 5.7, middle and bottom rows).

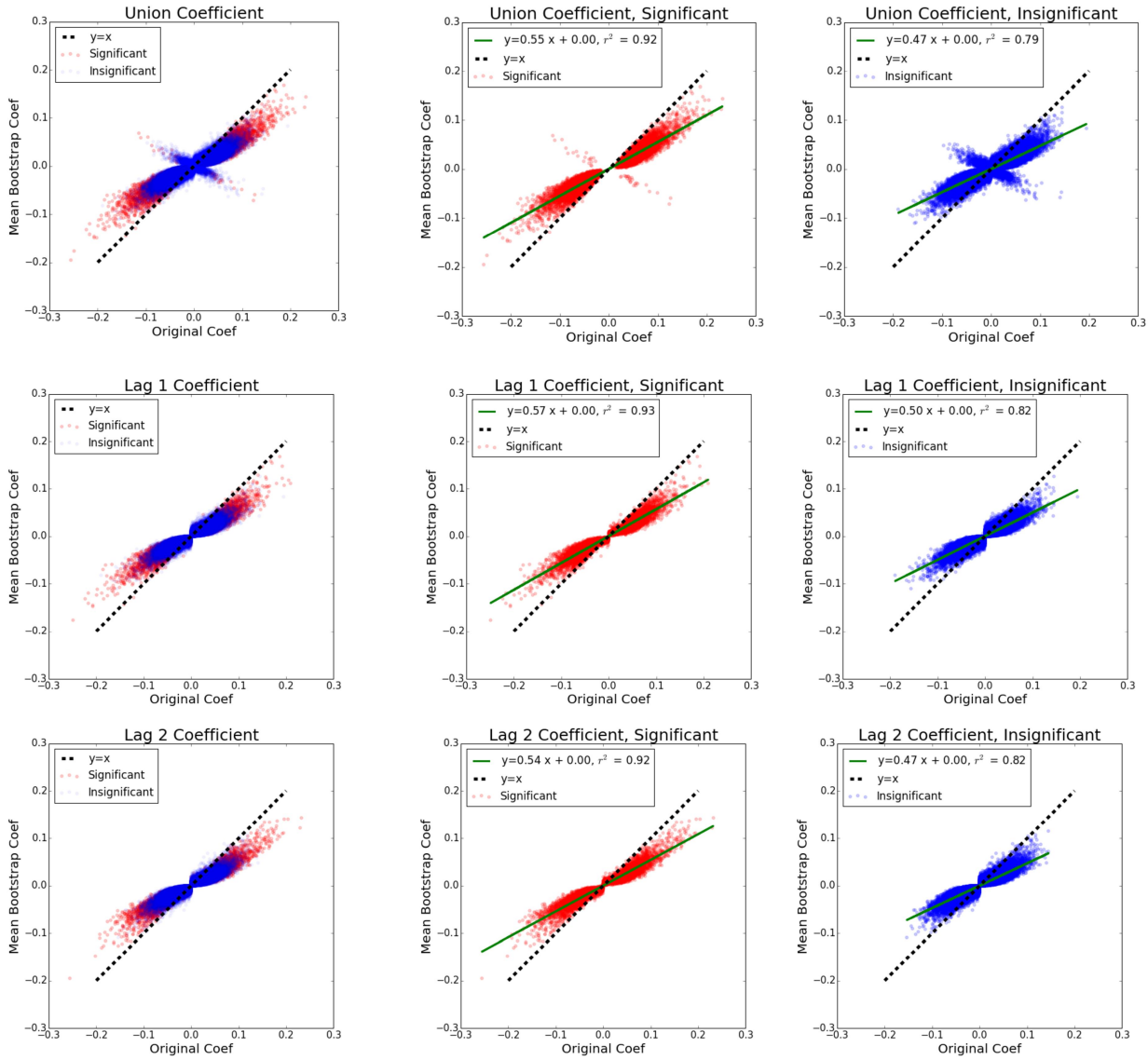

Figure 5.7: **Comparison of Mean of Coefficients across Bootstrap Networks with Coefficient from Original Fit, BETS-2.** In each plot, the x-axis lists the coefficient from the fit on the original complete data, while the y-axis lists the coefficient averaged over all the bootstrap networks. "Union Coefficient" refers to the Lag 1 and Lag 2 Coefficients. The red dots indicate significance at FDR 0.05 in the **original fit**, not in the bootstrap procedure used by BETS-2. Blue indicates insignificance.

In summary, there is high concordance between the coefficients inferred in fitting on the original data and the coefficients averaged from fits across multiple bootstrap resamples. The bootstrapping has the effect of shrinking the mean coefficient and generally not flipping the sign of the coefficient.

5.2.3 Validation

Literature-curated Network

We compared the two BETS-1 and BETS-2 networks to the 95-edge literature-curated network (Section 3.2.3).For both BETS-1 and BETS-2, we find **no** edges shared with the literature-curated network.

BETS-1 has only 1 out of its 2098 edges in which both the causal gene is in one of the literature-curated network's 29 causal genes, and the effect gene is in one of the literature-curated networks' 32 effect genes. This edge is $FOS \rightarrow CXCL8$. However, this is not actually a validated edge in the regulatory network.

BETS-2 has only 5 out of its 31945 edges in which both the causal gene is in one of the literature-curated network's 29 causal genes, and the effect gene is in one of the literature-curated networks' 32 effect genes. These edges are $FOX \rightarrow AFP$, $FOS \rightarrow CXCL8, FOS \rightarrow SGK1, NR4A1 \rightarrow SGK1, FOS \rightarrow FGG$. Again, none of these are actually validated edges in the regulatory network.

The low rate of intersection can be explained by the low number of genes present in the data. Again, upon limiting to the literature-curated network only to those genes that were present in our data, we reduced it from 95 to 7 edges: $JUN \rightarrow AFP$, $NR3C1 \rightarrow SGK1$, $NR3C1 \rightarrow CXCL8$, $NR3C1 \rightarrow AFP$, $NR3C1 \rightarrow FGG$, $NR3C1 \rightarrow BAX$, and $NR3C1 \rightarrow VIPR1$. 6 of these 7 involve $NR3C1$, which, while it encodes the Glucocorticoid Receptor, is only very lowly differentially expressed at an FDR of 0.2 (Section 3.2.1). As a nuclear receptor, its mechanism of action is not by increasing its own expression levels but binding the glucocorticoid drug when it

enters and then affecting transcription within the nucleus.

Over-expression Data

In this section we attempt. In particular, we rely on the signs (positive or negative) of our inferred causal relationships. We wish to emphasize that this analysis cannot directly be performed by those of BETS' competitors which are based on mutual information or decision trees, because those methods only output a positive measure of the causal relation between genes without information about the sign. Thus, sign information improves our method's and other vector autoregression methods' interpretability relative to other frameworks.

First, we inspected the BETS-1 and BETS-2 networks for edges involving the transcription factors used for the over-expression datasets (Table 5.6). We find in both networks, $TFCP2L1$ has the most out-edges (59 in BETS-1, 122 in BETS-2), and much more than the corresponding in-edges (1 in BETS-1 and 15 in BETS-2). However, the other 9 transcription factors had much less out-edges: 0 in BETS-1 and 1 to 17 in BETS-2. In fact, 7 of the 9 had more in-edges than out-edges. This is problematic because Transcription Factors should be regulated other genes and so have more out-edges, not regulated by other genes and have more in-edges.

	BETS-1		BETS-2	
	Edges From	Edges To	Edges From	Edges To
CEBPB	0	0	7	10
CEBPD	0	0	7	15
FOSL2	0	0	0	7
FOXO1	0	0	0	9
FOXO3	0	0	0	17
KLF6	0	0	10	13
KLF9	0	0	1	10
KLF15	0	0	0	0
POU5F1	0	0	10	10
TFCP2L1	59	1	122	15
All genes	2098	2098	31551	31551

Table 5.6: **Frequencies of Transcription Factor-related Edges for BETS-1 and BETS-2 Networks.** Transcription Factors for which we had an over-expressed dataset are listed. "Edges from" indicates edges where the gene is the cause; e.g. for $CEBPB$ all edges of form $CEBPB \rightarrow g$. "Edges to" indicates edges where the gene is the effect; e.g. for $CEBPB$, all edges of form $g \rightarrow CEBPB$. "All genes" simply lists all the edges of the network.

We inspected those edges inferred with $TFCP2L1$ as the causal gene. We analyzed the relationship between the inferred edge and the log-fold change. We fit the log-fold change as a function of the edge's sign (as the indicator variables "Is-Positive-Edge", "Is-Negative-Edge", and "Is-Edge"), and vice-versa. Logistic regression was used when the output variable was the edge sign. We use a significance threshold of 0.1.

In BETS-1, we find only a significant relationship between "Is-Edge" and the absolute log2-fold change ($p = 0.02$ for "Is-Edge" as the output in a logistic fit, and $p = 0.03$ for "Abs-log2fc" as the output in a linear fit) (Tables 5.8, 5.8). Oddly, the regression coefficient is the **opposite** of what we would expect: -1.4 when Is-Edge is the output and -0.099 when Abs-log2fc is the output. This **violates** our expectations that if $TFCP2L1 \rightarrow g$, then after we overexpress $TFCP2L1$, g would have a larger change from the original dataset than a gene with no edge from $TFCP2L1$ (Tables 5.10, 5.10).

Model	Fit	P-value of m
Log2fc = m Is-Pos-Edge +b	Log2fc = -0.066 Is-Pos-Edge 0.13	0.454
Log2fc = m Is-Neg-Edge +b	Log2fc = 0.085 Is-Neg-Edge -0.13	0.4
Abs-Log2fc = m Is-Edge +b	Abs-log2FC = -0.099* Is-Edge + 0.40	**0.026**

Table 5.7: **Edge to Log-Fold Change Regression Results, BETS-1** For a given gene g, "Log2fc" refers to the log2 fold-change of g's expression in the $TFCP2L1$ over-expressed dataset relative to the original data. "Abs-Log2fc" is the absolute value of "Log2fc". "Is-Pos-Edge" is 1 if there is an edge $TFCP2L1 \rightarrow g$ in the BETS-1 inferred network with positive coefficient and 0 otherwise. "Is-Neg-Edge" is 1 if there is an edge $TFCP2L1 \rightarrow g$ in the BETS-1 inferred network with negative coefficient and 0 otherwise. "Is-Edge" is 1 if there is an edge $TFCP2L1 \rightarrow g$ in the BETS-1 inferred network and 0 otherwise. Bold p-values indicate significance at $\alpha = 0.1$.

Model	Fit	P-value of m
Is-Pos-Edge = logit(m log2fc+b)	Is-Pos-Edge = logit(-0.24 * log2fc + -4.5)	0.452
Is-Neg-Edge = logit(m log2fc +b)	Is-Neg=Edge = logit(-0.3 * log2fc 4.7)	0.39
Is-Edge ∼logit(m abs-log2fc +b)	Edge = logit(-1.4* abs-log2fc - 3.3)	**0.02**

Table 5.8: **Log-Fold Change to Edge Regression Results, BETS-1** For a given gene g, "Log2fc" refers to the log2 fold-change of g's expression in the $TFCP2L1$ over-expressed dataset relative to the original data. "Abs-Log2fc" is the absolute value of "Log2fc". "Is-Pos-Edge" is 1 if there is an edge $TFCP2L1 \rightarrow g$ in the BETS-1 inferred network with positive coefficient and 0 otherwise. "Is-Neg-Edge" is 1 if there is an edge $TFCP2L1 \rightarrow g$ in the BETS-1 inferred network with negative coefficient and 0 otherwise. "Is-Edge" is 1 if there is an edge $TFCP2L1 \rightarrow g$ in the BETS-1 inferred network and 0 otherwise. Bold p-values indicate significance at $\alpha = 0.1$.

In BETS-2, we find a significant relationship between "Is-Positive-Edge" and the log2-fold change ($p = 0.00019$ when "Is-Pos-Edge" is the output, in a logistic fit, and $p = 0.00021$ when log-fold change is the output, in a linear fit). Oddly, the regression coefficient is the **opposite** of what we would expect: -0.68 when Is-Pos-Edge is the output and -0.22 when log2-fold change is the output. We also find a significant relationship between "Is-Edge" and the absolute log2-fold change ($p = 0.096$ for "Is-Edge" as the output, in a logistic fit, and $p = 0.096$ for "Abs-log2fc" as the output in a linear fit). Here, the regression cefficients are positive: 0.37 when "Is-Edge" is the output and 0.052 when Abs-log2fc is the output. This **concords** with our expectations that if $TFCP2L1 \rightarrow g$, then after we overexpress $TFCP2L1$, g would have a larger change from the original dataset than a gene with no edge from $TFCP2L1$ (Tables 5.10, 5.10).

In summary, in BETS-2, we find a significant positive relationship between whether the edge $TFCP2L1 \rightarrow g$ exists in the network, and the gene g's log-fold change in the $TFCP2L1$ over-expression dataset. This relationship has the right sign because we would expect a larger change from genes affected by the over-expressed factor. We also find significant negative relationships between whether there is a positive edge $TFCP2L1 \rightarrow g$ in the network, and the gene g's log-fold change. This relationship has the opposite sign from what we would expect because we would expect a gene that is activated by $TFCP2L1$ to have a more positive, not negative, fold-change relative to other genes. Finally, in BETS-1, we find an unexpected negative relationship between whether the edge $TFCP2L1 \rightarrow g$ exists in the network, and the gene g's log-fold change. This has the wrong sign because the genes cause by $TFCP2L1$ appear to change less in the held-out dataset.

Model	Fit	P-value of m
Log2fc = m Is-Pos-Edge +b	Log2fc = -0.22 Is-Pos-Edge 0.12	**0.00021**
Log2fc = m Is-Neg-Edge +b	Log2fc = 0.099 Is-Neg-Edge -0.13	0.17
Abs-Log2fc = m Is-Edge +b	Abs-log2FC = 0.052* Is-Edge + 0.40	**0.096**

Table 5.9: **Edge to Log-Fold Change Regression Results, BETS-2** For a given gene g, "Log2fc" refers to the log2 fold-change of g's expression in the $TFCP2L1$ over-expressed dataset relative to the original data. "Abs-Log2fc" is the absolute value of "Log2fc". "Is-Pos-Edge" is 1 if there is an edge $TFCP2L1 \rightarrow g$ in the BETS-2 inferred network with positive coefficient and 0 otherwise. "Is-Neg-Edge" is 1 if there is an edge $TFCP2L1 \rightarrow g$ in the BETS-2 inferred network with negative coefficient and 0 otherwise. "Is-Edge" is 1 if there is an edge $TFCP2L1 \rightarrow g$ in the BETS-2 inferred network and 0 otherwise. Bold p-values indicate significance at $\alpha = 0.1$.

Model	Fit	P-value of m
Is-Pos-Edge = logit(m log2fc $+b$)	Is-Pos-Edge = logit(-0.68 * log2fc + -3.8)	**0.00019**
Is-Neg-Edge = logit(m log2fc$+b$)	Is-Neg-Edge = logit(0.42 * log2fc 4.0)	0.17
Is-Edge $\sim$logit(m abs-log2fc $+b$)	Edge = logit(0.37* abs-log2fc - 3.2)	**0.096**

Table 5.10: **Log-Fold Change to Edge Regression Results, BETS-2** For a given gene g, "Log2fc" refers to the log2 fold-change of g's expression in the $TFCP2L1$ over-expressed dataset relative to the original data. "Abs-Log2fc" is the absolute value of "Log2fc". "Is-Pos-Edge" is 1 if there is an edge $TFCP2L1 \rightarrow g$ in the BETS-2 inferred network with positive coefficient and 0 otherwise. "Is-Neg-Edge" is 1 if there is an edge $TFCP2L1 \rightarrow g$ in the BETS-2 inferred network with negative coefficient and 0 otherwise. "Is-Edge" is 1 if there is an edge $TFCP2L1 \rightarrow g$ in the BETS-2 inferred network and 0 otherwise. Bold p-values indicate significance at $\alpha = 0.1$.

Chapter 6

Conclusion

This thesis reviews and then develops one effective solution for causal network inference from gene expression time series. We review 20 methods across 6 existing frameworks. We also both review and perform comparisons of these approaches on real and simulated datasets. We discuss why causal inference from high-dimensional time series is especially challenging both statistically and computationally.

We next develop an effective and scalable approach to statistical inference of causal networks, Bootstrap Elastic net regression from Time Series (BETS). BETS' key innovations are 1) to use the Elastic Net penalty for regularization, which handles correlated genes while preserving sparsity, and 2) to use the bootstrap frequency to rank edges rather than the coefficient. We further use a global null to solve the problem of declaring statistical significance from bootstrap frequencies. We evaluate BETS extensively against the DREAM4 100-gene challenge, and implement multiple approaches from distinct frameworks in-house for the sake of reproducible research. BETS ranks 3rd in AUROC out of 17 and 6th in AUPR out of 22, is one of the fastest methods compared to methods of similar performance, and outperforms all other Vector Autoregression approaches. Additionally, it provides signs to its relationship, unlike its competitors that are based on Decision Tree and Mutual Information.

We apply BETS to the GR data and infer two networks using lags of 1 and 2. We then analyze these networks for enrichment of various gene and edge types, finding enrichment of edges that involve transcription factors, immune, and metabolic genes as causes. We find high intersection between our two networks. We find that the out-degree distributions are heavily right-skewed while in-degree distributions are much more concentrated at low values, consistent with a smaller set of causal genes controlling regulation of a larger set of effect genes. We then specifically investigate the relative association of transcription factors with immunity and metabolism. We compare the bootstrap procedure with the original procedure and find high correlation in the coefficient averaged over bootstrap fits and the coefficient from the original fit. The bootstrap fits have an effect of shrinking the coefficients by a factor of ~ 0.5.

Finally, we assess the validity of our network on two sources of data: a literature-curated network and a dataset in which transcription factors were over-expressed. The literature-curated network had only 7 of 95 edges involving only genes within our set of 2768 differentially expressed genes for analysis. This is in part because the network involved mostly transcription factors, which tended not be differentially expressed. We found no intersections between the BETS-1 and BETS-2 networks and the literature-curated network. However, we did find associations between our network edges and log-fold changes in the over-expression datasets. Some associations had the correct sign, while others had the opposite sign. Such signed analysis is a unique and interpretable feature of vector autoregression that competitor methods like mutual information and decision tree cannot perform.

Going forward, we are working on validating our network on trans-expression Quantitative Trait Loci from the Genotype Tissue Expression Consortium, in the same manner as in our previous work [27]. We expect that the network to find enrichment of associations just as in our previous work, due to similarity between our current procedure and previous procedure (Section 5.2.2). We would also like to

run the methods dynGENIE3 and tl-CLR which were published only this year and shown to be two of the fastest and most effective network inference methods (Section 2). We would like to assess concordance between networks inferred by dynGENIE3, tl-CLR, and BETS. Finally, we would like to model gene trajectories across time as more flexible Gaussian Processes. Unlike linear Vector Autoregressive processes, GPs can handle nonuniform spacing of timepoints, variation across replicates, and nonstationarity. We would develop a framework for causality between these responses, inspired by work in the medical timeseries domain [48].

Appendix A

Supplementary Tables

Transcription Factor	**Annotation**	**IM Effects**	**Immune Portion of IM Effects**
NR1D2		4	0.75
FOS	Immune, Associated with GR	5	0.75
NR4A1	Associated with GR	2	0.5
TSC22D3		2	0.5
ATF3	Metabolism	2	0.5
TFCP2L1	Direct target of GR	4	0

Table A.1: **Transcription Factors and Immune-Metabolic Effects in BETS-1 Network.** Annotations are based on Section 3.3.

Transcription Factor	**Annotation**	**IM Effects**	**Immune Portion of IM Effects**
HOXB5		2	1
NR1D2		9	0.63
ZNF114		33	0.6
NR4A1	Associated with GR	25	0.58
SP6		2	0.5
BHLHE40		2	0.5
KLF10		3	0.5
ATOH8		3	0.5
TEF		4	0.5
TSC22D3		7	0.5
ZNF404		7	0.5
HES1	Metabolism	23	0.5
E2F1		2	0.5
SNAI2		10	0.5
FOS	Immune, Associated with GR	42	0.47
ATF3	Metabolism	18	0.44
FOSL1		5	0.4
ZBED2		13	0.33
ID2		3	0.33
ID4		4	0.25
TFCP2L1	Direct target of GR	9	0.25

Table A.2: **Transcription Factors and Immune-Metabolic Effects in BETS-2 Network.** Annotations are based on Section 3.3.

Bibliography

[1] Äijö, T., and Lähdesmäki, H. Learning gene regulatory networks from gene expression measurements using non-parametric molecular kinetics. *Bioinformatics 25*, 22 (2009), 2937–2944.

[2] Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., et al. Gene ontology: tool for the unification of biology. *Nature Genetics 25*, 1 (2000), 25–29.

[3] Bansal, M., Gatta, G. D., and Di Bernardo, D. Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. *Bioinformatics 22*, 7 (2006), 815–822.

[4] Beal, M. J., Falciani, F., Ghahramani, Z., Rangel, C., and Wild, D. L. A bayesian approach to reconstructing genetic regulatory networks with hidden factors. *Bioinformatics 21*, 3 (2004), 349–356.

[5] Bonneau, R., Reiss, D. J., Shannon, P., Facciotti, M., Hood, L., Baliga, N. S., and Thorsson, V. The inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets de novo. *Genome biology 7*, 5 (2006), R36.

[6] Bühlmann, P., Kalisch, M., and Meier, L. High-dimensional statistics with a view toward applications in biology. *Annual Review of Statistics and Its Application 1* (2014), 255–278.

[7] Cain, D. W., and Cidlowski, J. A. Immune regulation by glucocorticoids. *Nature Reviews Immunology* (2017).

[8] Consortium, G. O. Gene ontology consoritum's curated list of immune genes, 2014. Accessed: 2017-04-22.

[9] Consortium, U. Uniprot: the universal protein knowledgebase. *Nucleic acids research 45*, D1 (2016), D158–D169.

[10] Faith, J. J., Hayete, B., Thaden, J. T., Mogno, I., Wierzbowski, J., Cottarel, G., Kasif, S., Collins, J. J., and Gardner, T. S. Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. *PLoS biology 5*, 1 (2007), e8.

[11] Geer, E. B., Islam, J., and Buettner, C. Mechanisms of glucocorticoid-induced insulin resistance: focus on adipose tissue function and lipid metabolism. *Endocrinology and Metabolism Clinics of North America 43*, 1 (2014), 75–102.

[12] Geurts, P., et al. dyngenie3: dynamical genie3 for the inference of gene networks from time series expression data. *Scientific reports 8*, 1 (2018), 3384.

[13] Granger, C. Testing for causality. *Journal of Economic Dynamics and Control 2* (1980), 329 – 352.

[14] Hartemink, A. J., Gifford, D. K., Jaakkola, T. S., Young, R. A., et al. Using graphical models and genomic expression data to statistically validate models of genetic regulatory networks. In *Pacific symposium on biocomputing* (2001), vol. 6, p. 266.

[15] Hecker, M., Lambeck, S., Toepfer, S., van Someren, E., and Guthke, R. Gene regulatory network inference: Data integration in dynamic modelsa review. *Biosystems 96*, 1 (2009), 86 – 103.

[16] Hlavácková-Schindler, K., and Bouzari, H. Granger lasso causal models in higher dimensions-application to gene expression regulatory networks. In *ECML/PKDD 2013 workshop scalable decision making: uncertainty, imperfection, deliberation (SCALE)* (2013).

[17] Hoerl, A. E., and Kennard, R. W. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics 12*, 1 (1970), 55–67.

[18] Hooper, S. D., Boué, S., Krause, R., Jensen, L. J., Mason, C. E., Ghanim, M., White, K. P., Furlong, E. E., and Bork, P. Identification of tightly regulated groups of genes during drosophila melanogaster embryogenesis. *Molecular systems biology 3*, 1 (2007), 72.

[19] Huynh-Thu, V. A., and Sanguinetti, G. Combining tree-based and dynamical systems for the inference of gene regulatory networks. *Bioinformatics 31*, 10 (2015), 1614–1622.

[20] Irrthum, A., Wehenkel, L., Geurts, P., et al. Inferring regulatory networks from expression data using tree-based methods. *PloS one 5*, 9 (2010), e12776.

[21] Lèbre, S. Inferring dynamic genetic networks with low order independencies. *Statistical applications in genetics and molecular biology 8*, 1 (2009), 1–38.

[22] Leek, J. T., and Storey, J. D. Capturing heterogeneity in gene expression studies by surrogate variable analysis. *PLOS Genetics 3*, 9 (09 2007), 1–12.

[23] Lim, N., Şenbabaoğlu, Y., Michailidis, G., and dAlché Buc, F. Okvar-boost: a novel boosting algorithm to infer nonlinear dynamics and interactions in gene regulatory networks. *Bioinformatics 29*, 11 (2013), 1416–1423.

[24] Liu, Z.-P. Reverse engineering of genome-wide gene regulatory networks from gene expression data. *Current Genomics 16*, 1 (02 2015), 3–22.

[25] Lopes, M., and Bontempi, G. Experimental assessment of static and dynamic algorithms for gene regulation inference from time series expression data. *Frontiers in Genetics 4* (2013), 303.

[26] Lozano, A. C., Abe, N., Liu, Y., and Rosset, S. Grouped graphical granger modeling for gene expression regulatory networks discovery. *Bioinformatics 25*, 12 (2009), i110–i118.

[27] Lu, J. Improved methods for causal inference and experimental prioritization in gene regulatory networks. Master's thesis, Princeton University, 5 2017. A Junior two-semester thesis.

[28] Madar, A., Greenfield, A., Vanden-Eijnden, E., and Bonneau, R. Dream3: Network inference using dynamic context likelihood of relatedness and the inferelator. *PLoS ONE 5*, 3 (2010), e9803.

[29] Marbach, D., Schaffter, T., Mattiussi, C., and Floreano, D. Generating realistic in silico gene networks for performance assessment of reverse engineering methods. *Journal of computational biology 16*, 2 (2009), 229–239.

[30] Margolin, A. A., Nemenman, I., Basso, K., Wiggins, C., Stolovitzky, G., Dalla Favera, R., and Califano, A. Aracne: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. In *BMC bioinformatics* (2006), vol. 7, BioMed Central, p. S7.

[31] McDowell, I. C., Barrera, A., Hong, L. K., Leichter, S. M., Majoros, B., Dumitrascu, B., Luo, K., D Ippolito, A. M., Song, L., Safi, A., Vockley, C. M., Lu, J., Kocak, D. D., Bartelt, L. C., Gersbach, C. A., Hartemink, A. J., Engelhardt, B. E., Crawford, G. E., and Reddy, T. E. Highly coordinated dynamics of the genomic response to glucocorticoids. *in submission* (2017).

[32] Meinshausen, N., and Bühlmann, P. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 72*, 4 (2010), 417–473.

[33] Meyer, P. E., Kontos, K., Lafitte, F., and Bontempi, G. Information-theoretic inference of large transcriptional regulatory networks. *EURASIP journal on bioinformatics and systems biology 2007* (2007), 8–8.

[34] Meyer, P. E., Lafitte, F., and Bontempi, G. minet: Ar/bioconductor package for inferring large transcriptional networks using mutual information. *BMC bioinformatics 9*, 1 (2008), 461.

[35] Mukhopadhyay, N. D., and Chatterjee, S. Causality and pathway search in microarray time series experiment. *Bioinformatics 23*, 4 (2007), 442.

[36] Opgen-Rhein, R., and Strimmer, K. Learning causal networks from systems biology time course data: an effective model selection procedure for the vector autoregressive process. *BMC Bioinformatics 8*, 2 (2007), 1.

[37] Penfold, C. A., Shifaz, A., Brown, P. E., Nicholson, A., and Wild, D. L. Csi: a nonparametric bayesian approach to network inference from multiple perturbed time series gene expression data. *Statistical applications in genetics and molecular biology 14*, 3 (2015), 307–310.

[38] Penfold, C. A., and Wild, D. L. How to infer gene networks from expression profiles, revisited. *Interface Focus 1*, 6 (2011), 857–870.

[39] Peters, J. *Causality: Lecture Notes*. ETH Zurich, ETH Zurich, 2015.

[40] Pillich, R. T., Chen, J., Rynkov, V., Welker, D., and Pratt, D. Ndex: a community resource for sharing and publishing of biological networks. *Protein Bioinformatics: From Protein Modifications and Networks to Proteomics* (2017), 271–301.

[41] Pramila, T., Wu, W., Miles, S., Noble, W. S., and Breeden, L. L. The forkhead transcription factor hcm1 regulates chromosome segregation genes and fills the s-phase gap in the transcriptional circuitry of the cell cycle. *Genes & development 20*, 16 (2006), 2266–2278.

[42] Pratt, D., Chen, J., Welker, D., Rivas, R., Pillich, R., Rynkov, V., Ono, K., Miello, C., Hicks, L., Szalma, S., et al. Ndex, the network data exchange. *Cell systems 1*, 4 (2015), 302–305.

[43] Rau, A., Jaffrézic, F., Foulley, J.-L., and Doerge, R. W. An empirical bayesian method for estimating biological networks from temporal microarray data. *Statistical Applications in Genetics and Molecular Biology 9*, 1 (2010).

[44] Reddy, T. E., Gertz, J., Crawford, G. E., Garabedian, M. J., and Myers, R. M. The hypersensitive glucocorticoid response specifically regulates period 1 and expression of circadian genes. *Molecular and Cellular Biology 32*, 18 (09 2012), 3756–3767.

[45] Reddy, T. E., Pauli, F., Sprouse, R. O., Neff, N. F., Newberry, K. M., Garabedian, M. J., and Myers, R. M. Genomic determination of the glucocorticoid response reveals unexpected mechanisms of gene regulation. *Genome Research 19*, 12 (2009), 2163–2171.

[46] Rhen, T., and Cidlowski, J. A. Antiinflammatory action of glucocorticoids new mechanisms for old drugs. *New England Journal of Medicine 353*, 16 (2005), 1711–1723. PMID: 16236742.

[47] Robinson, M. D., McCarthy, D. J., and Smyth, G. K. edger: a bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics 26*, 1 (01 2010), 139–140.

[48] Schulam, P., and Saria, S. Reliable decision support using counterfactual models. In *Advances in Neural Information Processing Systems* (2017), pp. 1697–1708.

[49] Shipp, L. E., Lee, J. V., Yu, C.-Y., Pufall, M., Zhang, P., Scott, D. K., and Wang, J.-C. Transcriptional regulation of human dual specificity protein phosphatase 1 (dusp1) gene by glucocorticoids. *PLoS ONE 5*, 10 (2010), e13754.

[50] Shojaie, A., and Michailidis, G. Discovering graphical granger causality using the truncating lasso penalty. *Bioinformatics 26*, 18 (09 2010), i517–i523.

[51] Spencer, S. J., and Tilbrook, A. The glucocorticoid contribution to obesity. *Stress 14*, 3 (2011), 233–246.

[52] Subramanian, A., Tamayo, P., Mootha, V. K., Mukherjee, S., Ebert, B. L., Gillette, M. A., Paulovich, A., Pomeroy, S. L., Golub, T. R., Lander, E. S., and Mesirov, J. P. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences 102*, 43 (2005), 15545–15550.

[53] Tam, G. H., Chang, C., and Hung, Y. S. Application of granger causality to gene regulatory network discovery. In *IEEE 6th International Conference on Systems Biology (ISB)* (2012), ISB '12.

[54] Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* (1996), 267–288.

[55] Traxler, M. F., Chang, D.-E., and Conway, T. Guanosine 3?, 5?-bispyrophosphate coordinates global gene expression during glucose-lactose diauxie in escherichia coli. *Proceedings of the National Academy of Sciences of the United States of America 103*, 7 (2006), 2374–2379.

[56] Wang, Z., Gerstein, M., and Snyder, M. Rna-seq: a revolutionary tool for transcriptomics. *Nature reviews. Genetics 10*, 1 (01 2009), 57–63.

[57] Yao, S., Yoo, S., and Yu, D. Prior knowledge driven granger causality analysis on gene regulatory network discovery. *BMC Bioinformatics 16* (2015), 273.

[58] Young, W. C., Raftery, A. E., and Yeung, K. Y. Fast bayesian inference for gene regulatory networks using scanbma. *BMC systems biology 8*, 1 (2014), 47.

[59] Zhang, H.-M., Chen, H., Liu, W., Liu, H., Gong, J., Wang, H., and Guo, A.-Y. Animaltfdb: a comprehensive animal transcription factor database. *Nucleic Acids Research 40*, Database issue (01 2012), D144–D149.

[60] Zhu, J., Chen, Y., Leonardson, A. S., Wang, K., Lamb, J. R., Emilsson, V., and Schadt, E. E. Characterizing dynamic changes in the human blood transcriptional network. *PLoS Computational Biology 6*, 2 (2010).

[61] Zoppoli, P., Morganella, S., and Ceccarelli, M. Timedelay-aracne: Reverse engineering of gene networks from time-course data by an information theoretic approach. *BMC bioinformatics 11*, 1 (2010), 154.

[62] Zou, C., and Feng, J. Granger causality vs. dynamic bayesian network inference: a comparative study. *BMC Bioinformatics 10*, 1 (2009), 122.

[63] Zou, H., and Hastie, T. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 67*, 2 (2005), 301–320.