A Penalized Regression Model for the Joint Estimation of eQTL Associations and Gene Network Structure

Micol Marchetti-Bowick
Machine Learning Department
Carnegie Mellon University
micolmb@cs.cmu.edu

Abstract

Background: A critical task in the study of biological systems is understanding how gene expression is regulated within the cell. This problem is typically divided into multiple separate tasks, including performing eQTL mapping to identify SNP-gene relationships and estimating gene network structure to identify gene-gene relationships.

Aim: In this work, we pursue a holistic approach to discovering the patterns of gene regulation in the cell. We present a new method for jointly performing eQTL mapping and gene network estimation while encouraging a transfer of information between the two tasks.

Data: We evaluate our approach on both synthetic data and on a real yeast eQTL dataset that contains $1,157$ SNP genotypes and $1,409$ gene expression measurements for 114 yeast samples.

Methods: To construct a unified model for jointly performing eQTL mapping and gene network inference, we formulate the problem as a multiple-output regression task in which we aim to learn the regression coefficients while simultaneously estimating the conditional independence relationships among the set of response variables. The approach we develop uses structured sparsity penalties to encourage the sharing of information between the regression coefficients and the output network in a mutually beneficial way. Our model, *inverse-covariance-fused lasso*, is formulated as a biconvex optimization problem that we solve via alternating minimization. We derive new, efficient optimization routines to solve each convex sub-problem that are based on existing state-of-the-art methods.

Results: We demonstrate the value of our approach by applying our method to both simulated data and a real yeast eQTL dataset data. Experimental results demonstrate that our approach outperforms a large number of existing methods on the recovery of the true sparse structure of both the eQTL associations and the gene network.

Conclusions: We show that inverse-covariance-fused lasso can be used to perform joint eQTL mapping and gene network estimation on a yeast dataset, yielding more biologically coherent results than previous work. Furthermore, the same problem setting appears in many different applications, and therefore our model can be deployed in a wide range of domains.

1 Introduction

A critical task in the study of biological systems is understanding how gene expression is regulated within the cell. Although this problem has been studied extensively over the past few decades, it has recently gained momentum due to the rapid advancement in techniques for high-throughput data acquisition. Within this broad task, two sub-problems of particular interest are (a) understanding how various genetic loci regulate gene expression, a problem known as eQTL mapping (Rockman and Kruglyak, 2006), and (b) determining which genes have a direct influence on the expression of other genes, a problem known as gene network estimation (Gardner and Faith, 2005). Prior work on learning regulatory associations has largely treated eQTL mapping and gene network estimation as completely separate problems.

In this work, we pursue a holistic approach to discovering the patterns of gene regulation in the cell by integrating eQTL mapping and gene network estimation into a single model and performing joint inference. Specifically, given a dataset that contains both genotype information for a set of single nucleotide polymorphisms (SNPs) and mRNA expression measurements for a set of genes, we aim to simultaneously learn the SNP-gene and gene-gene relationships. The key element of our approach is that we enable the transfer of knowledge between these two tasks in order to yield more accurate solutions to both problems. In particular, we assume that two genes that are tightly linked in a regulatory network are likely to be associated with similar sets of SNPs in an eQTL map, and vice versa. This allows us to use information about gene-gene relationships to learn more accurate eQTL associations, and similarly to use information about SNP-gene relationships to learn a more accurate gene network.

We construct a unified model for this problem by formulating it as a multiple-output regression task in which we jointly estimate the regression coefficients and the inverse covariance structure among the response variables. Specifically, given SNPs $x = (x_1, \ldots, x_p)$ and genes $y = (y_1, \ldots, y_q)$, our goal is to regress y on x and simultaneously estimate the inverse covariance of y. Under this model, the matrix of regression coefficients encodes the SNP-gene relationships in the eQTL map, whereas the inverse covariance matrix captures the gene-gene relationships in the gene network. In order to ensure that information is transferred between the two components of the model, we incorporate a regularization penalty that explicitly encourages pairs of genes that have a high weight in the inverse covariance matrix to also have similar regression coefficient values. This structured penalty enables the two estimates to learn from one another as well as from the data.

A large number of techniques have been developed to address eQTL mapping and gene network estimation in isolation. The traditional approach to eQTL mapping is to examine each SNP-gene pair independently and perform a univariate statistical test to determine whether an association exists between the two. Recently, a series of more complex multivariate models have been utilized for this task in order to jointly capture the effects of several SNPs on one gene (Michaelson et al., 2010). Going one step further, a few approaches have been developed that jointly model multiple SNPs and multiple genes (Kim and Xing, 2009; Kim et al., 2012). Conversely, the simplest method for gene network inference is to construct a graph in which each edge is weighted by the marginal correlation between the corresponding genes. Another related method that has recently gained popularity, known equivalently as Gaussian graphical model estimation or inverse covariance estimation, produces a network in which the edges are instead weighted by the conditional correlations (Dempster, 1972). Other approaches include using Bayesian networks or ordinary differential equations to model the relationships between genes. (Marbach et al., 2010).

In this work, we will use a multiple-output (also called multi-task) regression model to perform eQTL mapping and incorporate a Gaussian graphical model over the genes in order to simultaneously infer the gene network. However, it is well known that model estimation is challenging in the

high-dimensional setting in which we have more variables in the model than samples from which to learn. This is the precisely the setting frequently encountered in genomics, where we see datasets with at most a few hundred samples but as many as 1 million SNPs and thousands of genes. As a result, we will incorporate several regularization penalties in order to facilitate model estimation.

The challenge of high-dimensional data has already inspired a remarkable number of useful methods for penalized multi-task regression. Parameter estimation can be improved by leveraging sparse regularizers to encourage the outputs to share an underlying representation. For example, mixed-norm penalties such as the $\ell_{1,2}$ or $\ell_{1,\infty}$ norm can be used to encourage all outputs to have nonzero coefficients for the same set of inputs (Obozinski et al., 2008). In other cases, structured sparsity can be used to encourage similarities only among responses that are known to be related. Examples of such approaches include the group lasso (Yuan and Lin, 2006), which encourages groups of related outputs to have nonzero coefficients for the same subset of inputs, and the graph-guided fused lasso (Kim and Xing, 2009), which encourages pairs of outputs that are connected in a graph to have similar coefficient values. These penalties leverage prior knowledge of the relationships among the outputs (e.g. group or graph structures) to enforce similarity constraints.

The principal limitation of using such structured sparsity in multi-task regression is that it typically requires extensive prior knowledge of the relationships among the outputs, which is not always readily available. To circumvent this, another class of models have been developed that jointly learn the regression coefficients along with the output dependency structure (Rothman et al., 2010; Zhang and Yeung, 2010; Lee and Liu, 2012; Sohn and Kim, 2012; Rai et al., 2012). However, none of these approaches use a regularization penalty to explicitly encourage shared structure between the estimate of the regression parameters and the output network. Furthermore, the majority of these methods do not learn the covariance structure of the outputs y but rather the conditional covariance of the outputs given the inputs $y \,|\, x$. This can be interpreted as the covariance of the noise matrix, and does not capture the true relationships between outputs.

In high-dimensional settings, inverse covariance estimation models can also benefit from penalties that induce sparsity or encourage a particular structure. For example, Tan et al. (2014) use a row-column overlap norm penalty to encourage a network structure with hubs. When the goal is to learn a series of related networks for different conditions or time points, regularization can be used to encourage similarities between the networks (Mohan et al., 2014; Peterson et al., 2015).

Here we present a novel approach that jointly performs eQTL mapping and gene network inference while simultaneously leveraging structured sparsity. In particular, our work has three significant contributions: (1) we construct a novel multiple-output regression model that jointly estimates the regression coefficients and the output network while encouraging shared structure; (2) we develop an efficient alternating minimization procedure for our model, which has a biconvex objective, and derive nontrivial extensions to two state-of-the-art optimization techniques to solve each sub-problem; (3) we perform synthetic and real data experiments to demonstrate that our approach leads to more accurate estimation of the sparsity pattern of the regression coefficients, more accurate estimation of the sparse network structure, and a lower prediction error than baseline methods. To the best of our knowledge, there are no existing approaches that jointly estimate regression coefficients and output relationships while also incorporating regularization penalties to encourage the explicit sharing of information between the two estimates.

The remainder of this article is organized as follows. We begin by providing some background in Section 2, and then present our model and optimization technique in Sections 3 and 4. Next we demonstrate the value of our approach in Sections 5 and 6 by applying our model to both synthetic and real data, and comparing the results to several baselines. Finally, we conclude in Section 7 by discussing the implications of our findings.

2 Background

Before introducing our approach, we provide some background on the problems of penalized multiple-output regression and sparse inverse covariance estimation, which will form the building blocks of our unified model.

In what follows, we assume X is an n-by-p dimensional matrix of SNP genotypes, which we also call *inputs*, and Y is an n-by-q dimensional matrix of gene expression values, which we also call *outputs*. Here n is the number of samples, p is the number of SNPs, and q is the number of genes. The element $x_{ij} \in \{0, 1, 2\}$ represents the genotype value of sample i at SNP j, encoded as 0 for two copies of the minor allele, 1 for one copy of the minor allele, and 2 for two copies of the minor allele. Similarly $y_{ik} \in \mathbb{R}$ represents the expression value of sample i in gene k. We assume that the expression values for each gene are mean-centered.

2.1 Multiple-Output Lasso

Given input matrix X and output matrix Y, the standard ℓ_1-penalized multiple-output regression problem, also known as the multi-task lasso (Tibshirani, 1996), is given by

$$\min_B \ \tfrac{1}{n}\|Y - XB\|_F^2 + \lambda\|B\|_1 \tag{1}$$

where B is a p-by-q dimensional matrix and β_{jk} is the regression coefficient that maps SNP x_j to gene y_k. Here $\|\cdot\|_1$ is an ℓ_1 norm penalty that induces sparsity among the estimated coefficients, and λ is a regularization parameter that controls the degree of sparsity. The objective function given above is derived from the penalized negative log likelihood of a multivariate Gaussian distribution, assuming $y \,|\, x \sim \mathcal{N}(x^T B, \varepsilon^2 I)$ where we let $\varepsilon^2 = 1$ for simplicity. Although this problem is formulated in a multiple-output framework, the ℓ_1 norm penalty merely encourages sparsity, and does not enforce any shared structure between the regression coefficients of different outputs. As a result, the objective function given in (1) decomposes into q independent regression problems.

2.2 Graph-Guided Fused Lasso

Given a weighted graph $W \in \mathbb{R}^{q \times q}$ that encodes a set of pairwise relationships among the outputs, we can modify the regression problem by imposing an additional fusion penalty that encourages genes y_k and y_m to have similar parameter vectors $\beta_{\cdot k}$ and $\beta_{\cdot m}$ when the weight of the edge connecting them is large. This problem is known as the graph-guided fused lasso (Kim et al., 2009; Kim and Xing, 2009; Chen et al., 2010) and is given by

$$\begin{aligned}\min_B \ & \tfrac{1}{n}\|Y - XB\|_F^2 + \lambda\|B\|_1 \\ & + \gamma \textstyle\sum_{(k,m)} |w_{km}| \cdot \|\beta_{\cdot k} - \operatorname{sgn}(w_{km})\beta_{\cdot m}\|_1\end{aligned} \tag{2}$$

Here the ℓ_1 norm penalty again encourages sparsity in the estimated coefficient matrix. In contrast, the second penalty term, known as a graph-guided fusion penalty, encourages similarity among the regression parameters for all pairs of outputs. The weight of each term in the fusion penalty is dictated by $|w_{km}|$, which encodes the strength of the relationship between y_k and y_m. Furthermore, the sign of w_{km} determines whether to encourage a positive or negative relationship between parameters; if $w_{km} > 0$ (i.e. genes y_k and y_m are positively correlated), then we encourage $\beta_{\cdot k}$ to be equal to $\beta_{\cdot m}$, but if $w_{km} < 0$ (i.e. genes y_k and y_m are negatively correlated), we encourage $\beta_{\cdot k}$ to be equal to $-\beta_{\cdot m}$. If $w_{km} = 0$, then genes y_k and y_m are unrelated, and so we don't fuse their respective regression coefficients.

2.3 Sparse Inverse Covariance Estimation

In the graph-guided fused lasso model defined in (2), the graph W must be known ahead of time. However, it is also possible to *learn* a network over the set of genes. One way to do this is to estimate their pairwise conditional independence relationships. If we assume $y \sim \mathcal{N}(\mu, \Sigma)$, where we let $\mu = 0$ for simplicity, then these conditional independencies are encoded in the inverse covariance matrix, or precision matrix, defined as $\Theta = \Sigma^{-1}$. We can obtain a sparse estimate of the precision matrix using the graphical lasso (Friedman et al., 2008) given by

$$\min_{\Theta} \ \tfrac{1}{n}\text{tr}(Y^T Y\, \Theta) - \log\det(\Theta) + \lambda\|\Theta\|_1 \tag{3}$$

This objective is again derived from the penalized negative log likelihood of a Gaussian distribution, where this time the ℓ_1 penalty term encourages sparsity among the entries of the precision matrix.

3 The Inverse-Covariance-Fused Lasso

In this section, we introduce a new approach for jointly estimating the coefficients in a multiple-output regression problem and the edges of a network over the regression outputs. We apply this technique to the problem of simultaneously learning an eQTL map and a gene regulatory network from genome (SNP) data and transcriptome (gene expression) data. Although we focus exclusively on this application, the same problem formulation appears in other domains as well, such as the task of modeling the dependencies between a set of economic indicators and the stock prices of various companies while also understanding the relationships among the companies.

3.1 A Joint Regression and Network Estimation Model

Given SNPs $x \in \mathbb{R}^p$ and genes $y \in \mathbb{R}^q$, in order to jointly model the n-by-p regression parameter matrix B and the q-by-q inverse covariance matrix Θ, we begin with two core modeling assumptions,

$$x \sim \mathcal{N}(0, T) \tag{4}$$

$$y \,|\, x \sim \mathcal{N}(x^T B, E) \tag{5}$$

where T is the covariance of x and E is the conditional covariance of $y \,|\, x$. Given the above model, we can also derive the marginal distribution of y. To do this, we first use the fact that the marginal distribution $p(y)$ is Gaussian.[1] We can then use the law of total expectation and the law of total variance to derive the mean and covariance of y, as follows.

$$\mathbb{E}_y(y) = \mathbb{E}_x(\mathbb{E}_{y|x}(y|x)) = 0 \tag{6}$$

$$\mathbb{C}\text{ov}_y(y) = \mathbb{E}_x(\mathbb{C}\text{ov}_{y|x}(y|x)) + \mathbb{C}\text{ov}_x(\mathbb{E}_{y|x}(y|x)) = E + B^T T B \tag{7}$$

Using these facts, we conclude that the distribution of y is given by

$$y \sim \mathcal{N}(0, \Theta^{-1}) \tag{8}$$

where $\Theta^{-1} = E + B^T T B$ denotes the marginal covariance of y. This allows us to explicitly relate Θ, the inverse covariance of y, to B, the matrix of regression parameters. Lastly, we simplify our model by assuming $T = \tau^2 I_{p\times p}$ and $E = \varepsilon^2 I_{q\times q}$. With this change, the relationship between B and Θ^{-1} can be summarized as $\Theta^{-1} \propto B^T B$ because B is now the only term that contributes to the off-diagonal entries of Θ and hence to the inverse covariance structure among the genes.

[1] We refer the reader to Equation B.44 of Appendix B in Bishop (2006).

3.2 Estimating Model Parameters with a Fusion Penalty

Now that we have a model that explicitly captures B and Θ, we want to jointly estimate these parameters from the data while encouraging the relationship $\Theta^{-1} \propto B^T B$. To do this, we formulate our model as a convex optimization problem with an objective function of the form

$$\text{loss}_{y|x}(B) + \text{loss}_y(\Theta) + \text{penalty}(B, \Theta) \tag{9}$$

where $\text{loss}_{y|x}(B)$ is a loss function derived from the negative log likelihood of $y \mid x$, $\text{loss}_y(\Theta)$ is a loss function derived from the negative log likelihood of y, and $\text{penalty}(B, -\Theta)$ is a penalty term that encourages shared structure between the estimates of B and Θ.

Given n i.i.d. observations of x and y, let X be a matrix that contains one observation of x per row and let Y be a matrix that contains one observation of y per row. Then we define the inverse-covariance-fused lasso optimization problem as

$$\begin{aligned} \min_{B,\Theta} \; & \tfrac{1}{n}\|Y - XB\|_F^2 + \tfrac{1}{n}\text{tr}(Y^T Y\,\Theta) - \log\det(\Theta) \\ & + \lambda_1\|B\|_1 + \lambda_2\|\Theta\|_1 \\ & + \gamma \textstyle\sum_{(k,m)} |\theta_{km}| \cdot \|\beta_{\cdot k} + \text{sgn}(\theta_{km})\beta_{\cdot m}\|_1 \end{aligned} \tag{10}$$

This objective effectively boils down to a combination of problems (1) and (3), with the addition of a graph-guided fusion penalty from (2) to encourage transfer learning between the estimate of B and Θ. We deconstruct the above objective by describing the role of each term in the model:

- The first term $\frac{1}{n}\|Y - XB\|_F^2$ is the standard squared error loss for multiple-output regression, and is derived from the log likelihood of $y \mid x \sim \mathcal{N}(x^T B, \varepsilon^2 I)$. Its role is to encourage the coefficients B to map X to Y.

- The second term $\frac{1}{n}\text{tr}(Y^T Y\,\Theta) - \log\det(\Theta)$ is the loss function for inverse covariance estimation over Y, and is derived from the log likelihood of $y \sim \mathcal{N}(0, \Theta^{-1})$. Its role is to encourage the network Θ to reflect the partial correlations among the outputs.

- The third term $\lambda_1\|B\|_1 = \lambda_1 \sum_{j,k} |\beta_{jk}|$ is an ℓ_1 norm penalty over the matrix of regression coefficients that induces sparsity in B.

- The fourth term $\lambda_2\|\Theta\|_1 = \lambda_2 \sum_{k,m} |\theta_{km}|$ is an ℓ_1 norm penalty over the precision matrix that induces sparsity in Θ.

- The final term $\gamma \sum_{(k,m)} |\theta_{km}| \cdot \|\beta_{\cdot k} + \text{sgn}(\theta_{km})\beta_{\cdot m}\|_1$ is a graph-guided fusion penalty that encourages similarity between the coefficients of closely related outputs; specifically, when y_k and y_m have a positive partial correlation and $\beta_{jk} \neq \beta_{jm}$ for any j, it imposes a penalty proportional to $|\theta_{km}|$, and when y_k and y_m have a negative partial correlation and $\beta_{jk} \neq -\beta_{jm}$ for any j, it imposes a penalty proportional to $|\theta_{km}|$.[2]

[2]Note that θ_{km} is negatively proportional to the partial correlation between y_k and y_m, meaning that a negative value of θ_{km} indicates a positive partial correlation and a positive value of θ_{km} indicates a negative partial correlation. The partial correlation coefficient between y_j and y_k is given by $\rho_{jk} = -\theta_{jk}/\sqrt{\theta_{ii}\theta_{jj}}$ and is defined as the correlation between the errors of the best linear predictions of y_j and y_k when conditioned on covariates $\{y_m : m \neq j, k\}$ (see, e.g., Peng et al. (2009)). This explains why the sign is flipped in the fusion penalty in (10) relative to the one in (2).

A derivation of the two loss functions can be found in Appendix A. These come directly out of the modeling assumptions given in (5) and (8). The two ℓ_1 norm penalties induce sparsity in the estimates of B and Θ, which makes estimation feasible in the high-dimensional setting where $p, q > n$ and furthermore is necessary for interpreting the eQTL map and gene network. Lastly, we prove that the graph-guided fused lasso penalty encourages the structure $\Theta^{-1} \propto B^T B$, thereby linking the two estimates. Note that λ_1, λ_2, and γ are regularization parameters that determine the relative weight of each penalty term.

First we consider the optimization problem $\hat{\Theta} = \arg\min_\Theta f(\Theta) \equiv \text{tr}(B^T B\Theta) - \log\det(\Theta)$. We can solve this problem in closed form by taking the gradient $\nabla_\Theta f(\Theta) = B^T B - \Theta^{-1}$ and setting it to 0, which yields the solution $\hat{\Theta}^{-1} = B^T B$. This suggests that the penalty $\text{tr}(B^T B\Theta)$ encourages the desired structure, while the log determinant term enforces the constraint that Θ be positive semidefinite, which is necessary for Θ to be a valid inverse covariance matrix.

Instead of directly using this penalty in our model, we demonstrate its equivalence to the graph-guided fused lasso penalty shown in (2) when we substitute $-\Theta$ for W. We compare the trace penalty, denoted TRP, and the graph-guided fused lasso penalty, denoted GFL, below.

$$\text{TRP}(B, \Theta) = \text{tr}(B^T B\Theta) = \sum_{k=1}^{q}\sum_{m=1}^{q} \theta_{km} \cdot \beta_{\cdot k}^T \beta_{\cdot m} \tag{11}$$

$$\text{GFL}(B, -\Theta) = \sum_{k=1}^{q}\sum_{m=1}^{q} |\theta_{km}| \cdot \|\beta_{\cdot k} + \text{sgn}(\theta_{km})\beta_{\cdot m}\|_1 \tag{12}$$

We show how these penalties are equivalent by considering three different cases.

- **Case of $\theta_{km} = 0$.** When $\theta_{km} = 0$, the corresponding term in (11) becomes $0 \cdot \beta_{\cdot k}^T \beta_{\cdot m} = 0$ and the corresponding term in (12) becomes $0 \cdot \|\beta_{\cdot k} + \text{sgn}(\theta_{km})\beta_{\cdot m}\|_1 = 0$. Therefore there is nothing linking $\beta_{\cdot k}$ and $\beta_{\cdot m}$ when $\theta_{km} = 0$ in either penalty.

- **Case of $\theta_{km} < 0$.** When $\theta_{km} < 0$, the corresponding term in (11) is minimized when $\beta_{\cdot k}^T \beta_{\cdot m}$ is large and positive, which occurs when $\beta_{\cdot k}$ and $\beta_{\cdot m}$ point in the same direction. Similarly, the corresponding term in (12) becomes $|\theta_{km}| \cdot \|\beta_{\cdot k} - \beta_{\cdot m}\|_1$ and the penalty is minimized when $\beta_{\cdot k} = \beta_{\cdot m}$. Therefore when θ_{km} is negative, both penalties encourage similarity between $\beta_{\cdot k}$ and $\beta_{\cdot m}$ with strength proportional to the magnitude of θ_{km}.

- **Case of $\theta_{km} > 0$.** When $\theta_{km} > 0$, the corresponding term in (11) is minimized when $\beta_{\cdot k}^T \beta_{\cdot m}$ is large and negative, which occurs when $\beta_{\cdot k}$ and $\beta_{\cdot m}$ point in opposite directions. Similarly, the corresponding term in (12) becomes $|\theta_{km}| \cdot \|\beta_{\cdot k} + \beta_{\cdot m}\|_1$ and the penalty is minimized when $\beta_{\cdot k} = -\beta_{\cdot m}$. Therefore when θ_{km} is positive, both penalties encourage similarity between $\beta_{\cdot k}$ and $-\beta_{\cdot m}$ with strength proportional to the magnitude of θ_{km}.

We choose to use the graph-guided fused lasso penalty instead of the trace penalty because it more strictly enforces the relationship between B and Θ^{-1} by fusing the regression parameter values of highly correlated genes.

3.3 Relationship to Other Methods

There are currently two existing approaches that jointly estimate regression coefficients and network structure: multivariate regression with covariance estimation (MRCE), from Rothman et al. (2010), and conditional Gaussian graphical models (CGGMs), originally from Sohn and Kim (2012) and

further developed by Wytock and Kolter (2013) and Yuan and Zhang (2014). In this section, we describe how our approach differs from these others.

All three methods (including ours) assume that the inputs X and outputs Y are related according to the basic linear model $Y = XB + E$, where E is a matrix of Gaussian noise. However, each approach imposes a different set of additional assumptions on top of this, which we discuss below.

MRCE: This method assumes that $E \sim \mathcal{N}(0, \Omega^{-1})$, which leads to $Y \mid X \sim \mathcal{N}(XB, \Omega^{-1})$. MRCE estimates B and Ω by solving the following objective:

$$\begin{aligned} \min_{B,\Omega} \; & \tfrac{1}{n}\text{tr}((Y - XB)^T(Y - XB)\,\Omega) \\ & - \log\det(\Omega) + \lambda_1\|B\|_1 + \lambda_2\|\Omega\|_1 \end{aligned} \tag{13}$$

It's very important to note that Ω is the conditional inverse covariance of $Y \mid X$, which actually corresponds to the inverse covariance of the noise matrix E rather than the inverse covariance of the output matrix Y. We therefore argue that Ω doesn't capture any patterns that are shared with the regression coefficients B, since by definition Ω encodes the structure in Y that cannot be explained by XB.

CGGM: This approach makes an initial assumption that X and Y are jointly Gaussian with the following distribution:

$$\begin{pmatrix} X \\ Y \end{pmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \Gamma & \Lambda \\ \Lambda^T & \Omega \end{bmatrix}\right)$$

In this formulation, the distribution of $Y \mid X$ is given by $Y \mid X \sim \mathcal{N}(-X\Lambda\Omega^{-1}, \Omega^{-1})$. This corresponds to the reparameterization of B as $-\Lambda\Omega^{-1}$, where Ω is the conditional inverse covariance matrix and Λ represents the "direct" influence of X on Y. CGGMs estimate Λ and Ω by solving the following optimization problem, where sparsity penalties are applied to Λ and Ω instead of B and Ω as was the case in (13):

$$\begin{aligned} \min_{\Lambda,\Omega} \; & \tfrac{1}{n}\text{tr}((Y + X\Lambda\Omega^{-1})^T(Y + X\Lambda\Omega^{-1})\,\Omega) \\ & - \log\det(\Omega) + \lambda_1\|\Lambda\|_1 + \lambda_2\|\Omega\|_1 \end{aligned} \tag{14}$$

Here the meaning of Ω has not changed, and it once again represents the inverse covariance of the noise matrix.

ICLasso: Our method implicitly assumes two underlying models: $Y \mid X \sim \mathcal{N}(XB, I)$ and $Y \sim \mathcal{N}(0, \Theta^{-1})$. In this case, Θ represents the marginal inverse covariance of Y rather than the conditional inverse covariance of $Y \mid X$, which was captured by Ω in (13) and (14). The optimization problem in (10) is obtained by combining the loss functions derived from the log likelihood of each model and then incorporating sparsity penalties over B and Θ and an additional graph-guided fusion penalty to encourage shared structure.

Both MRCE and CGGMs have two important drawbacks that are not shared by our approach. First, both of these methods estimate Ω, the precision matrix of the noise term, rather than Θ, the precision matrix of the outputs Y. This means that they are estimating a network over the outputs in which each edge represents a partial correlation after conditioning on both the xs and all other ys. Second, neither method incorporates a structured sparsity penalty that explicitly encourages shared structure between the network and the regression coefficients. In fact, it would not make sense for these methods to apply a joint penalty over B and Ω because, as discussed above, we wouldn't expect these parameters to have any shared structure. By comparison, our method learns the true output network Θ and uses a graph-guided fused lasso penalty to explicitly encourage outputs that are closely related in Θ to have similar parameter values in B.

4 Optimization via Alternating Minimization

In this section we present an efficient algorithm to solve the inverse-covariance-induced fused lasso problem defined in (10). We start by rewriting the fusion penalty as follows:

$$\begin{aligned}
&\mathrm{GFL}(B, -\Theta) \\
&= \gamma \textstyle\sum_{(k,m)} |\theta_{km}| \cdot \|\beta_{\cdot k} + \mathrm{sgn}(\theta_{km})\beta_{\cdot m}\|_1 \\
&= \gamma \textstyle\sum_{(k,m)} \max\{\theta_{km}, 0\} \cdot \|\beta_{\cdot k} + \beta_{\cdot m}\|_1 \\
&\quad + \gamma \textstyle\sum_{(k,m)} \max\{-\theta_{km}, 0\} \cdot \|\beta_{\cdot k} - \beta_{\cdot m}\|_1,
\end{aligned}$$

from which it is clear that GFL is a bi-convex function, meaning that with fixed B, it is convex in Θ, and vice versa. Thus, upon defining

$$\begin{aligned}
g(B) &= \tfrac{1}{n}\|Y - XB\|_F^2 + \lambda_1\|B\|_1 \\
h(\Theta) &= \tfrac{1}{n}\mathrm{tr}(Y^T Y\,\Theta) - \log\det(\Theta) + \lambda_2\|\Theta\|_1,
\end{aligned}$$

we can rewrite the original objective as

$$\min_{B,\Theta}\ g(B) + h(\Theta) + \mathrm{GFL}(B, -\Theta). \tag{15}$$

The function g is the usual lasso formulation in (1), the function h is the graphical lasso formulation in (3), and the graph-guided fusion penalty couples the two problems. Fortunately, since GFL is bi-convex, we can solve the joint problem (15) using an alternating minimization strategy. Next we leverage and extend state-of-the-art convex optimization routines to solve each convex sub-problem.

Fix Θ, Minimize B. When Θ is fixed, minimizing the objective over B reduces to the well-known graph-guided fused lasso problem,

$$f_\Theta(B) = g(B) + \mathrm{GFL}(B, -\Theta), \tag{16}$$

which we optimize using the proximal-average proximal gradient descent (PA-PG) algorithm from Yu (2013). This algorithm is very simple. On each iteration, we first take a gradient step of the form $B - \eta X^\top(XB - Y)$ using some small step size η. Then we compute the weighted average of the component proximal operators for each pair of outputs, where the prox that corresponds to pair (k, m) is given by:

$$\hat{B} = \arg\min_B\ \tfrac{1}{2\eta}\|B - Z\|_F^2 + \|\beta_{\cdot k} + \mathrm{sgn}(\theta_{km})\beta_{\cdot m}\|_1 \tag{17}$$

and the weight of this term is given by $|\theta_{km}|/\theta_{\mathrm{tot}}$ where $\theta_{\mathrm{tot}} = \sum_{(k,m)} |\theta_{k,m}|$. Due to the separability of (17) over the rows of B, we can solve for each $\beta_{j\cdot}$ independently. Furthermore, it's clear that for any $i \notin \{k, m\}$, we have $\beta_{ji} = z_{ji}$. Solving for the remaining elements β_{jk} and β_{jm} leads to the following two-dimensional subproblem:

$$\hat{\beta}_{jk}, \hat{\beta}_{jm} = \arg\min_{\beta_{jk},\beta_{jm}}\ \tfrac{1}{2\eta}(\beta_{jk} - z_{jk})^2 + (\beta_{jm} - z_{jm})^2 + |\beta_{jk} + \mathrm{sgn}(\theta_{km})\beta_{jm}|. \tag{18}$$

which can be solved in closed form. Therefore the full solution to the prox operator can be written compactly as follows, where $d_{km} = z_{\cdot k} + \mathrm{sgn}(\theta_{km})z_{\cdot m}$.

$$\begin{aligned}
&\hat{\beta}_{\cdot i} = z_{\cdot i} \quad \text{for } i \notin \{k, m\} \\
&\hat{\beta}_{\cdot k} = z_{\cdot k} - \mathrm{sgn}(d_{km}) \cdot \min\{\eta, \tfrac{1}{2}|d_{km}|\} \\
&\hat{\beta}_{\cdot m} = z_{\cdot m} - \mathrm{sgn}(\theta_{km}) \cdot \mathrm{sgn}(d_{km}) \cdot \min\{\eta, \tfrac{1}{2}|d_{km}|\}
\end{aligned}$$

From these formulas, we can see that β_{jk} and $-\text{sgn}(\theta_{km})\beta_{jm}$ are always "fused" towards each other. For example, when $\text{sgn}(\theta_{km}) < 0$, we want to push β_{jk} and β_{jm} towards the same value. In this case, the larger of z_{jk} and z_{jm} will be decremented and the smaller value will be incremented by the same quantity. In contrast, when $\text{sgn}(\theta_{km}) > 0$, the solution pushes β_{jk} towards $-\beta_{jm}$.

We summarize the procedure in Algorithm 1 of Appendix B. In practice, we use the accelerated version of the algorithm, PA-APG. Using the argument from Yu (2013), we can prove that this accelerated algorithm converges to an ϵ-optimal solution in at most $O(1/\epsilon)$ steps, which is significantly better than the converge rate of subgradient descent, which is $O(1/\sqrt{\epsilon})$.

Fix B, Minimize Θ. When B is fixed, minimizing the objective over Θ reduces to a variation of the well-known graphical lasso problem,

$$f_B(\Theta) = h(\Theta) + \text{GFL}(B, -\Theta), \tag{19}$$

which can be optimized by adapting the block coordinate descent (BCD) algorithm of Friedman et al. (2008). Indeed, we can rewrite the objective by introducing two $q \times q$ dimensional coefficient matrices U and L whose elements are defined as

$$U_{km} = \tfrac{1}{n} Y_{\cdot k}^\top Y_{\cdot m} + \lambda_2 + \gamma \|\beta_{\cdot k} + \beta_{\cdot m}\|_1 \tag{20}$$

$$L_{km} = \tfrac{1}{n} Y_{\cdot k}^\top Y_{\cdot m} - \lambda_2 - \gamma \|\beta_{\cdot k} - \beta_{\cdot m}\|_1. \tag{21}$$

Using this notation, we collect all linear terms involving $\Theta_+ := \max\{\Theta, 0\}$ and $\Theta_- := \max\{-\Theta, 0\}$ and reformulate the objective given in (19) as

$$\min_{\Theta} -\log\det(\Theta) + \langle \Theta_+, U\rangle - \langle \Theta_-, L\rangle. \tag{22}$$

The graphical lasso is a special case of the above problem in which $U = L$. In our case, U and L differ because of the structure of the GFL penalty. Nevertheless, we can derive a block coordinate algorithm for this more general setting.

First we dualize (22) to get the following problem:

$$\max_{L \le \Xi \le U} \log\det \Xi. \tag{23}$$

where $\Theta = \Xi^{-1}$. Then it can be shown that the diagonal of the covariance Ξ must attain the upper bound, i.e. we must have $\Xi_{jj} = U_{jj}\ \forall\ j = 1, \ldots, q$. Next, we perform block coordinate descent by cycling through each column (or row, due to symmetry) of Ξ. We denoted an arbitrary column of Ξ by ξ_j, with corresponding columns u_j and ℓ_j in U and L, respectively. Let $\tilde{\Xi}_j$ be the submatrix of Ξ obtained by deleting column j and row j. Then, by applying Schur's complement, maximizing (23) with respect to ξ_j with all other columns fixed amounts to:

$$\min_{\ell_j \le \xi_j \le u_j}\ \tfrac{1}{2}\xi_j^\top \tilde{\Xi}_j^{-1} \xi_j. \tag{24}$$

Dualizing again, with $\xi_j = -\tilde{\Xi}_j \alpha$, we obtain

$$\min_{\alpha}\ \tfrac{1}{2}\alpha^\top \tilde{\Xi}_j \alpha + u^\top \alpha_+ - \ell^\top \alpha_-, \tag{25}$$

which is essentially a lasso problem that we can solve using any known algorithm. Algorithm 2 of Appendix B outlines our procedure for solving (19). We use coordinate descent and apply a variant of the soft-thresholding operator to solve for each coordinate. This algorithm converges very quickly because there is no tuning of the step size, and each iteration involves only a matrix-vector product.

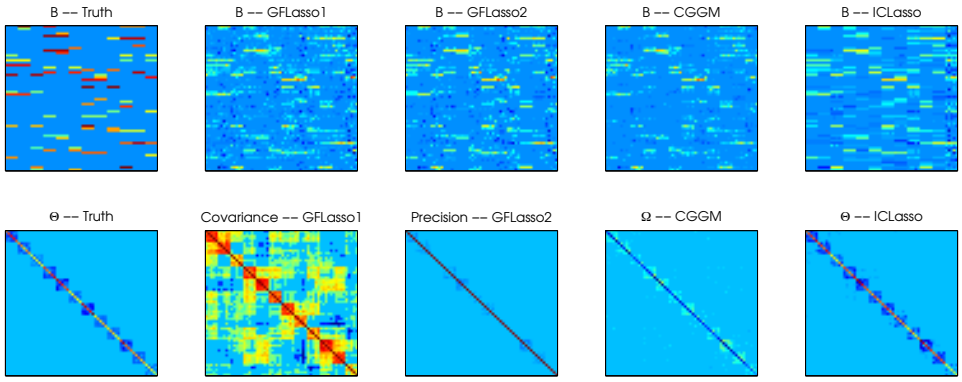

Figure 1: A comparison of results on a single synthetic dataset with $p = 60$, $q = 60$, and a block-structured output graph. The far left panel contains the ground truth for B and Θ. The remaining panels show the estimates of the regression coefficients for each method (top) along with the graph structure that was used or estimated by the method (bottom).

5 Simulation Study

We first evaluate our method on synthetic data with known values of B and Θ so that we can directly measure how well the the true parameter values are recovered. We compare our regression estimates to several baselines, including $\hat{B}$ standard multi-task lasso (Lasso), graph-guided fused lasso using a sparse covariance matrix as its graph (GFLasso1), graph-guided fused lasso using a sparse precision matrix as its graph (GFlasso2), sparse multivariate regression with covariance estimation (MRCE), and the conditional Gaussian graphical model (CGGM). We also compare our network estimates $\hat{\Theta}$ to the graphical lasso (GLasso). For each method, we selected hyperparameter values by minimizing the error on a held-out validation set.

For this analysis, we used two different synthetic data settings, each imposing a different network structure over the outputs:

- block-structured network: the outputs are divided into non-overlapping groups, and each group forms a fully connected sub-graph in the network
- tree-structured network: the outputs are related to one another according to a tree with a single root node and a branching factor of three, and all nodes share edges with their parents and children in the network

Each synthetic dataset is generated using $p = 60$ inputs, $q \in \{30, 60, 90, 120\}$ outputs, and $n = 50$ samples. After constructing a network over the outputs using one of the structures described above, we generate the sparsity pattern and parameter values for the coefficient matrix B. For the block-structured network, we choose a random 5% of inputs for each group, and assign positive weights between these inputs and all nodes in the group. We draw coefficient values according to $v_j \sim \text{Unif}(0.2, 0.8)$ and set $\beta_{jk} = v_j \ \forall \ k \in \text{group}$, such that all members of the group have the same coefficient value for these inputs. We also randomly select 5% additional inputs for each group that will be shared across all members of that group and one other group, and generate coefficients in the same way. For the tree-structured network, we choose a random 5% of inputs for each node, and assign a positive weight between those inputs and both the node and each of its children. We similarly generate coefficient values according to $v_j \sim \text{Unif}(0.2, 0.8)$ and set $\beta_{jk} = v_j \ \forall \ k \in \{\text{parent,children}\}$. Next we generate $X \sim \mathcal{N}(0, \Sigma)$ where $\Sigma_{jk} = 0.6^{|j-k|}$. Finally,

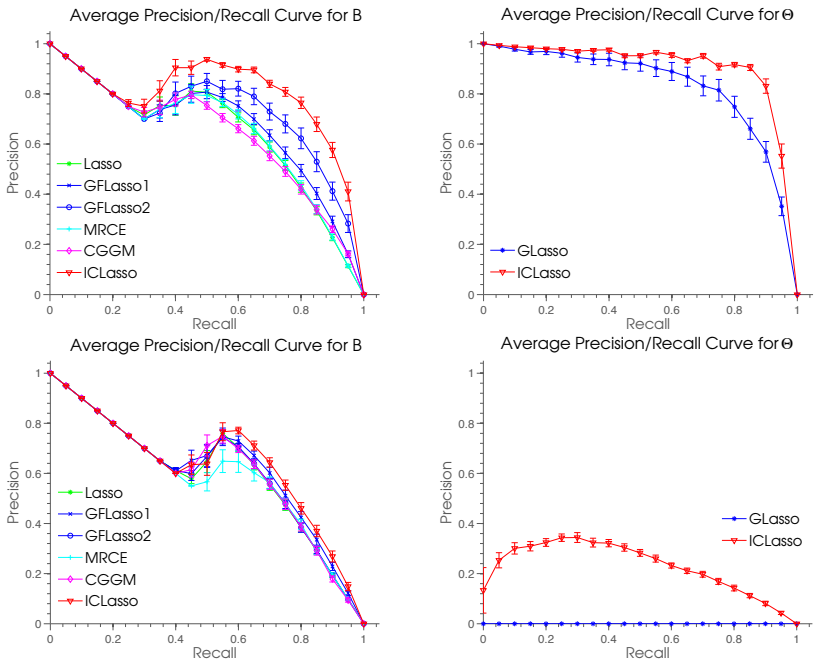

Figure 2: Average precision-recall curves for B (left) and Θ (right) for synthetic datasets with two different parameter settings: $p = 60$, $q = 120$, block-structured graph (top), and $p = 60$, $q = 60$, tree-structured graph (bottom).

given X and B, we generate $Y \sim \mathcal{N}(XB, I)$. We create multiple synthetic datasets by varying the network structure and the number of outputs. For each setting, we average our results over 15 synthetic datasets.

A synthetic data example is shown in Figure 1. The ground truth for both B and Θ is given in the far left panel. The next three columns show the estimated values of B for three competing methods, and the results of our method are shown on the far right. In this example, the drawbacks of each of the baseline methods are evident. The covariance matrix used for the network structure in GFLasso1 captures many spurious patterns in Y that don't correspond to true patterns in the regression map, which confuses the estimate of B. The precision matrix used for the network structure in GFLasso2 does not accurately capture the true inverse covariance structure because of the low signal-to-noise ration in Y. This prevents the fusion penalty from effectively influencing the estimate of B. Finally, although CGGM gets a reasonable estimate of the network, this structure is not explicitly enforced in B, which still leads to a poor estimate of the regression parameters. In contrast, the cleanest estimate of both $\hat{B}$ and $\hat{\Theta}$ comes from ICLasso.

Next we quantitatively evaluate how well each method is able estimate the sparsity structure of B and Θ. Since the hyperparameter values selected by minimizing the error on a validation set do not usually correspond to the values that lead to the most accurate sparsity, we calculate a precision-recall curve on top of $\hat{B}$ and $\hat{\Theta}$. To do this, we vary the threshold at which each parameter value is considered "zero" or "nonzero" and we calculate the precision and recall at each threshold. Figure 2 shows the precision-recall curves for B and Θ averaged over 15 datasets in two different parameter settings. The top two plots show results for a block-structured network with $q = 120$. Here it's evident that ICLasso significantly outperforms all of the baselines in its estimate of both the regression map and network structure. The bottom two plots show results for a tree-structured

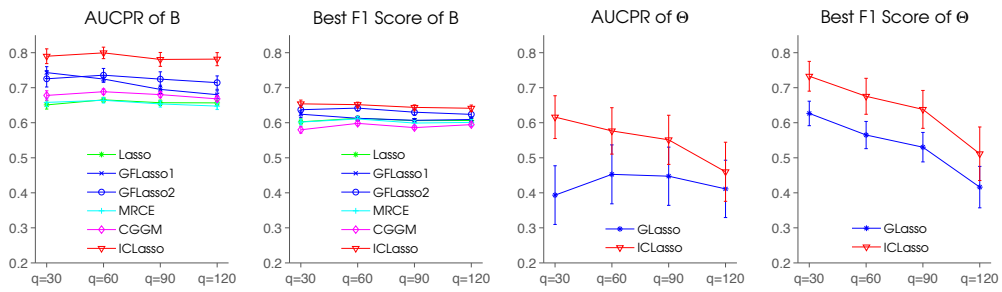

Figure 3: Area under the precision-recall curve and best F1 score over all regularization parameter values for B and Θ shown for different numbers of outputs q and averaged over 30 synthetic datasets generated with both graph types.

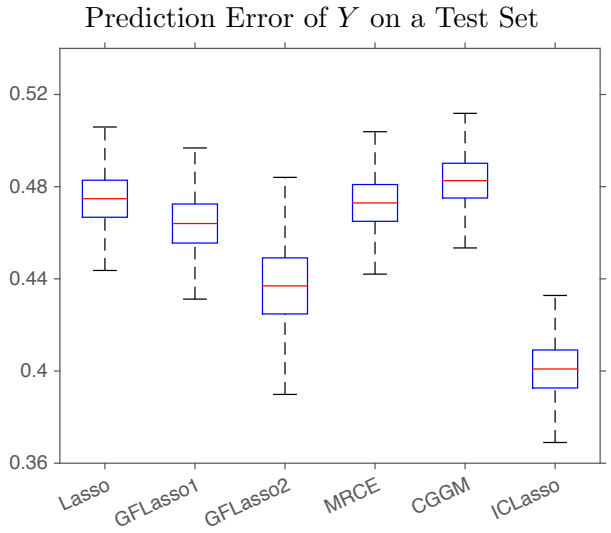

Figure 4: Test set prediction error on synthetic data with $p = 60$, $q = 120$, and a block-structured graph. Each box shows the mean (red), standard error interval (blue), and standard deviation (black) over 15 random datasets.

network with $q = 60$. Although ICLasso wins on B by a small margin, it's true advantage in this setting is its estimate of Θ. To further quantify these results, the the average area under the PR curve for B is 0.87 ± 0.01 for ICLasso and 0.79 ± 0.02 for GFLasso2, which is the closest baseline. The average area under the PR curve for Θ is 0.91 ± 0.01 for ICLasso and 0.82 ± 0.06 for GLasso.

Figure 3 shows the average area under the precision-recall curve and best F1 score (over all hyperparameter values) for B and Θ averaged over both network structures for each value of q. This shows that ICLasso consistently outperforms all baselines over multiple random datasets generated with multiple parameter settings. Finally, we examine the prediction error of each model when using $\hat{B}$ to predict Y from X. Figure 4 shows a box plot of the error values obtained in one parameter setting. This demonstrates that ICLasso obtains lower error than the other methods.

The results on synthetic data demonstrate that ICLasso performs significantly better than competing methods on a wide variety of metrics (accurate recovery of sparse structure in B, accurate recovery of sparse structure in Θ, and prediction of Y) when our data assumptions hold.

Table 1: Regression Error on Yeast Data

	density	training error	validation error
Lasso	1.65%	0.502	0.718
GFLasso	2.87%	0.392	0.715
ICLasso	6.88%	0.395	0.703

6 Yeast eQTL Mapping

In order to evaluate our approach in a real-world setting, we applied ICLasso to a yeast eQTL dataset from Brem and Kruglyak (2005) that consists of $2,956$ SNP genotypes and $5,637$ gene expression measurements across 114 yeast samples. Before running our method, we removed $1,799$ SNPs with duplicate genotypes and retained only the top 25% of genes with the highest variance in expression, leaving $p = 1,157$ SNPs and $q = 1,409$ genes in our analysis.

We used our approach to jointly perform eQTL mapping and gene network inference on the yeast dataset, treating the the SNPs as inputs X and the genes as outputs Y. We trained our model on 91 samples and used the remaining 23 samples as a validation set for tuning the hyperparameters. This yielded $\lambda_1 = 0.154$, $\lambda_2 = 0.157$, and $\gamma = 0.118$. Given the trained model, we read the eQTL associations from the regression coefficient matrix $\hat{B}$, which encodes SNP-gene relationships, and obtained the gene network from the inverse covariance matrix $\hat{\Theta}$, which encodes gene-gene relationships. In addition to ICLasso, we ran Lasso and GFlasso on the yeast data to obtain two additional estimates of B, and ran graphical lasso to obtain another estimate of Θ.[3]

Table 1 shows the density of $\hat{B}$ obtained with each method, along with the prediction error of Y on the training set and on the held-out validation set, which were calculated using $\|Y_{\text{train}} - X_{\text{train}}\hat{B}\|_F^2$ and $\|Y_{\text{valid}} - X_{\text{valid}}\hat{B}\|_F^2$, respectively. We chose not to sacrifice any data for a test set, but these results indicate that ICLasso achieves an equivalent or better fit to the training and validation sets than lasso and glasso.

6.1 Quantitative Analysis

Because the true yeast eQTLs and gene network structure are not known, there is no ground truth for this problem. We instead analyzed the output of each method by performing a series of enrichment analyses that together provide a comprehensive picture of the biological coherence of the results. An enrichment analysis uses gene annotations to identify specific biological processes, functions, or structures that are over-represented among a group of genes relative to the full set of genes that is examined (Subramanian et al., 2005). To evaluate our yeast data results, we performed three types of enrichment analyses: biological process and molecular function enrichment using annotations from the Gene Ontology (GO) database (Ashburner et al., 2000), and pathway enrichment using annotations from the Kyoto Encyclopedia of Genes and Genomes (KEGG) database (Kanehisa and Goto, 2000). We used a hypergeometric test to compute a p-value for each term, and then adjusted the values to account for multiple hypothesis testing. Significance was determined using an adjusted p-value cutoff of 0.01.

[3]The hyperparameters for each of these methods were selected with the same validation set, and we used $\lambda = 0.307$ for lasso, $\lambda = 0.231$ and $\gamma = 0.039$ for gflasso, and $\lambda = 0.471$ for glasso.

Table 2: GO and KEGG Enrichment Analysis on Yeast eQTL Map

	number of enriched terms			average increase	number of enriched SNPs			average increase
	GO-BP	GO-MF	KEGG		GO-BP	GO-MF	KEGG	
Lasso	1862	804	205	—	198	132	127	—
GFLasso	3499	1528	312	+ **77**%	286	211	172	+**47**%
ICLasso	8046	3147	1025	+**155**%	590	453	441	+**126**%

Table 3: GO and KEGG Enrichment Analysis on Yeast Gene Network

	number of enriched terms			average increase	number of enriched clusters			average increase
	GO-BP	GO-MF	KEGG		GO-BP	GO-MF	KEGG	
GLasso	173	77	31	—	14	12	11	—
ICLasso	321	127	41	+**61**%	29	26	22	+**108**%

We first analyzed $\hat{B}$ by performing a per-SNP enrichment analysis. For each SNP j, we used the nonzero elements in $\beta_{j\cdot}$ to identify the set of genes associated with the SNP. Next we performed GO and KEGG enrichment analyses on this group of genes by comparing their annotations to the full set of $1,409$ genes that we included in our study. We repeated this procedure for each SNP, and calculated the total number of terms that were enriched over all SNPs to obtain a global measure of enrichment for $\hat{B}$. In addition, we calculated the total number of SNPs that were enriched for at least one term in each category. These results are summarized in Table 2. It is evident that ICLasso vastly outperforms both GFLasso and Lasso on estimating the regression coefficients, since it has more than twice as many enriched terms in GO biological process, GO molecular function, and KEGG than either baseline method.

Next we used a similar approach to evaluate the structure present in $\hat{\Theta}$. We first obtained groups of genes by using spectral clustering to perform community detection among the genes based on the inferred network structure. After clustering the genes into 100 groups,[4] we performed GO and KEGG enrichment analyses on each cluster and calculated the total number of enriched terms along with the total number of clusters that were enriched for at least one term. These results are summarized in Table 3. Once again, our approach has more enrichment than the baseline in every category, which implies that the gene network estimated by ICLasso has a much more biologically correct structure than the network estimated by GLasso.

6.2 Qualitative Analysis

The quantitative results in Tables 2 and 3 indicate that, compared to other methods, our approach identifies more eQTLs that are associated with genes significantly enriched in certain biological processes and pathways. A more detailed examination of our results revealed that many of the enriched terms correspond to metabolic pathways, and that the eQTLs we identified agree with those discovered in a previous study that analyzed the effect of genetic variations on the yeast metabolome.

[4]We also clustered with 25, 50, and 200 groups and obtained similar results.

Breunig et al. (2014) identified the metabolite quantitative trait loci (mQTLs) for 34 metabolites and then examined each mQTL for the presence of metabolic genes in the same pathway as the linked metabolite. We found that 10 of these 34 metabolites were linked to metabolic genes where our identified eQTLs reside. For example, Breunig et. al. determined that the metabolite valine is linked to an mQTL in a region spanned by the ILV6 gene, which encodes a protein involved in valine biosynthesis. In our study, we also identified an eQTL located in ILV6. Moreover, we found that the eQTL in ILV6 is associated with 365 genes that are significantly enriched for pathways involved in the metabolism and biosynthesis of various amino acids. This is consistent with the fact that the metabolism and biosynthesis of amino acids in the cell needs to be coordinated.

Furthermore, our enrichment analysis shows that the eQTL-associated genes we identified are enriched for various metabolic pathways (e.g. sulfur, riboflavin, protein, starch, and sucrose metabolism, oxidative phosphorylation, glycolysis), as well as more general pathways, such as cell cycle pathways, and MAPK pathways. This is consistent with the roles of the mQTLs identified by Breunig et al. Interestingly, among these genes, SAM1, encoding an S-adenosylmethionine synthetase, is also among the eQTLs in our list. Our results show that the eQTL we found in SAM1 is associated with 252 genes that are enriched for cytoplasmic translation and ribosome functions, consistent with the fact that SAM is the methyl donor in most methylation reactions and is essential for DNA methylation of proteins, nucleic acids, and lipids (Roberts and Selker, 1995).

7 Conclusion

In this work, we propose a new model called the *inverse-covariance-induced fused lasso* which jointly estimates regression coefficients B and an output network Θ while using a graph-guided fused lasso penalty to explicitly encourage shared structure. Our model is formulated as a biconvex optimization problem, and we derive new, efficient optimization routines for each convex subproblem based on existing methods.

Our results on both synthetic and real data unequivocally demonstrate that our model achieves significantly better performance on recovery of the structure of B, recovery of the structure of Θ, and prediction error than all six baselines that we evaluated. In this paper, we demonstrated that our approach can effectively be used to perform joint eQTL mapping and gene network estimation on a yeast dataset, yielding more biologically coherent results than previous work. However, the same problem setting appears in many different applications, and the inverse-covariance-induced fused lasso model can therefore be effectively used within a wide range of domains.

References

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. (2000). Gene ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29.

Bishop, C. M. (2006). Pattern recognition. *Machine Learning*.

Brem, R. B. and Kruglyak, L. (2005). The landscape of genetic complexity across 5,700 gene expression traits in yeast. *Proceedings of the National Academy of Sciences*, 102(5):1572–1577.

Breunig, J. S., Hackett, S. R., Rabinowitz, J. D., and Kruglyak, L. (2014). Genetic basis of metabolome variation in yeast. *PLoS Genetics*, 10(3):e1004142.

Chen, X., Kim, S., Lin, Q., Carbonell, J. G., and Xing, E. P. (2010). Graph-structured multi-task regression and an efficient optimization method for general fused lasso. *arXiv preprint arXiv:1005.3579*.

Dempster, A. P. (1972). Covariance selection. *Biometrics*, pages 157–175.

Friedman, J., Hastie, T., and Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441.

Gardner, T. S. and Faith, J. J. (2005). Reverse-engineering transcription control networks. *Physics of Life Reviews*, 2(1):65–88.

Kanehisa, M. and Goto, S. (2000). Kegg: kyoto encyclopedia of genes and genomes. *Nucleic Acids Research*, 28(1):27–30.

Kim, S., Sohn, K.-A., and Xing, E. P. (2009). A multivariate regression approach to association analysis of a quantitative trait network. *Bioinformatics*, 25(12):i204–i212.

Kim, S. and Xing, E. P. (2009). Statistical estimation of correlated genome associations to a quantitative trait network. *PLoS Genetics*, 5(8):e1000587.

Kim, S., Xing, E. P., et al. (2012). Tree-guided group lasso for multi-response regression with structured sparsity, with an application to eqtl mapping. *The Annals of Applied Statistics*, 6(3):1095–1117.

Lee, W. and Liu, Y. (2012). Simultaneous multiple response regression and inverse covariance matrix estimation via penalized gaussian maximum likelihood. *Journal of Multivariate Analysis*, 111:241–255.

Marbach, D., Prill, R. J., Schaffter, T., Mattiussi, C., Floreano, D., and Stolovitzky, G. (2010). Revealing strengths and weaknesses of methods for gene network inference. *Proceedings of the National Academy of Sciences*, 107(14):6286–6291.

Michaelson, J. J., Alberts, R., Schughart, K., and Beyer, A. (2010). Data-driven assessment of eqtl mapping methods. *BMC Genomics*, 11(1):502.

Mohan, K., London, P., Fazel, M., Witten, D., and Lee, S.-I. (2014). Node-based learning of multiple gaussian graphical models. *The Journal of Machine Learning Research*, 15(1):445–488.

Obozinski, G. R., Wainwright, M. J., and Jordan, M. I. (2008). High-dimensional support union recovery in multivariate regression. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1217–1224.

Peng, J., Wang, P., Zhou, N., and Zhu, J. (2009). Partial correlation estimation by joint sparse regression models. *Journal of the American Statistical Association*, 104(486).

Peterson, C., Stingo, F. C., and Vannucci, M. (2015). Bayesian inference of multiple gaussian graphical models. *Journal of the American Statistical Association*, 110(509):159–174.

Rai, P., Kumar, A., and Daume, H. (2012). Simultaneously leveraging output and task structures for multiple-output regression. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3185–3193.

Roberts, C. J. and Selker, E. U. (1995). Mutations affecting the biosynthesis of s-adenosylmethionine cause reduction of dna methylation in neurospora crassa. *Nucleic Acids Research*, 23(23):4818–4826.

Rockman, M. V. and Kruglyak, L. (2006). Genetics of global gene expression. *Nature Reviews Genetics*, 7(11):862–872.

Rothman, A. J., Levina, E., and Zhu, J. (2010). Sparse multivariate regression with covariance estimation. *Journal of Computational and Graphical Statistics*, 19(4):947–962.

Sohn, K.-A. and Kim, S. (2012). Joint estimation of structured sparsity and output structure in multiple-output regression via inverse-covariance regularization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1081–1089.

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., et al. (2005). Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, 102(43):15545–15550.

Tan, K. M., London, P., Mohan, K., Lee, S.-I., Fazel, M., and Witten, D. (2014). Learning graphical models with hubs. *The Journal of Machine Learning Research*, 15(1):3297–3331.

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

Wytock, M. and Kolter, Z. (2013). Sparse gaussian conditional random fields: algorithms, theory, and application to energy forecasting. In *International Conference on Machine Learning (ICML)*, pages 1265–1273.

Yu, Y. (2013). Better approximation and faster algorithm using the proximal average. In *Advances in Neural Information Processing Systems (NIPS)*.

Yuan, M. and Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67.

Yuan, X.-T. and Zhang, T. (2014). Partial gaussian graphical model estimation. *IEEE Transactions on Information Theory*, 60(3):1673–1687.

Zhang, Y. and Yeung, D.-Y. (2010). A convex formulation for learning task relationships in multi-task learning. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence, UAI 2010*.

Appendix A Proofs

Derivation of loss function over B. Given the model $y \mid x \sim \mathcal{N}(x^T B, \varepsilon^2 I)$, we can estimate B by minimizing the negative log likelihood of $y \mid x$. The optimization objective is given below.

$$
\begin{aligned}
f(B) &= -\log \textstyle\prod_{i=1}^{n} p(y_i|x_i) = -\sum_{i=1}^{n} \log p(y_i|x_i) \\
&= -\textstyle\sum_{i=1}^{n} \log(|2\pi\varepsilon^2 I|^{-\frac{1}{2}} \exp\{-\frac{1}{2}(y_i - x_i^T B)^T(\varepsilon^2 I)^{-1}(y_i - x_i^T B)\}) \\
&= \tfrac{1}{2}\textstyle\sum_{i=1}^{n} \log\left(|2\pi E|\right) + \frac{1}{2\varepsilon^2}\sum_{i=1}^{n}(y_i - x_i^T B)^T(y_i - x_i^T B) \\
&\propto \textstyle\sum_{i=1}^{n}(y_i - x_i^T B)^T(y_i - x_i^T B) \\
&= \|Y - XB\|_F^2
\end{aligned}
$$

Derivation of loss function over Θ. Given the model $y \sim \mathcal{N}(0, \Theta^{-1})$, we can estimate Θ by minimizing the negative log likelihood of y. The optimization objective is given below.

$$
\begin{aligned}
f(\Theta) &= -\log \textstyle\prod_{i=1}^{n} p(y_i) = -\sum_{i=1}^{n} \log p(y_i) \\
&= -\textstyle\sum_{i=1}^{n} \log(|2\pi\Theta|^{\frac{1}{2}} \exp\{-\frac{1}{2} y_i^T \Theta y_i\}) \\
&= -\tfrac{1}{2}\textstyle\sum_{i=1}^{n} \log|2\pi\Theta| + \frac{1}{2}\sum_{i=1}^{n} y_i^T \Theta y_i \\
&\propto -n\log|\Theta| + \textstyle\sum_{i=1}^{n} y_i^T \Theta y_i \\
&= -n\log|\Theta| + \operatorname{tr}(Y^T Y \Theta)
\end{aligned}
$$

Appendix B Algorithms

Algorithm 1 PA-PG for Graph-Guided Fused Lasso

input: data X, Y, graph Θ, step size η
initialize: $B = 0$
repeat
  $B \leftarrow B - \eta X^\top (XB - Y)$
  for each edge (k, m) with $\theta_{km} \neq 0$ **do**
    $d_{km} \leftarrow \beta_{.k} + \mathrm{sgn}(\theta_{km})\beta_{.m}$
    $\beta_{.k} \leftarrow \beta_{.k} - (\theta_{km}/\theta_{\mathrm{tot}}) \cdot \min\{\eta, \frac{1}{2}|d_{km}|\}$
    $\beta_{.m} \leftarrow \beta_{.m} - (\theta_{km}/\theta_{\mathrm{tot}}) \cdot \min\{\eta, \frac{1}{2}|d_{km}|\}$
  end for
until convergence

Algorithm 2 BCD for the Generalized Graphical Lasso

input: coefficient matrices U, L
repeat
  for $j = 1$ **to** q **do**
    $\Xi_{jj} \leftarrow U_{jj}$
    $\xi \leftarrow \Xi_{\setminus j,j}$, $u \leftarrow U_{\setminus j,j}$, $\ell \leftarrow L_{\setminus j,j}$, $\tilde{\Xi} \leftarrow \Xi_{\setminus j,\setminus j}$
    repeat
      for $j = 1$ **to** $q - 1$ **do**
        $\delta \leftarrow \sum_{k \neq j} \tilde{\Xi}_{jk}\alpha_k$
        if $\delta \geq -l_j$ **then**
          $\alpha_j = (-\delta - l_j)/\tilde{\Xi}_{jj}$
        else if $\delta \leq -u_j$ **then**
          $\alpha_j = (-\delta - u_j)/\tilde{\Xi}_{jj}$
        else
          $\alpha_j = 0$
        end if
      end for
    until convergence
    $\Xi_{\setminus i,\setminus i} \leftarrow -\tilde{\Xi}\boldsymbol{\beta}$
  end for
until convergence
$\Theta = \Xi^{-1}$