Learning Modular Structures from Network Data and Node Variables

Elham Azizi ELHAM@BU.EDU
Bioinformatics Program, Boston University, Boston, MA 02215 USA

Edoardo M. Airoldi AIROLDI@FAS.HARVARD.EDU
Department of Statistics, Harvard University, Camrbdige, MA 02138 USA

James E. Galagan JGALAG@BU.EDU
Departments of Biomedical Engineering and Microbiology, Boston University, Boston, MA 02215 USA

Abstract

A standard technique for understanding underlying dependency structures among a set of variables posits a shared conditional probability distribution for the variables measured on individuals within a group. This approach is often referred to as module networks, where individuals are represented by nodes in a network, groups are termed modules, and the focus is on estimating the network structure among modules. However, estimation solely from node-specific variables can lead to spurious dependencies, and unverifiable structural assumptions are often used for regularization. Here, we propose an extended model that leverages direct observations about the network in addition to node-specific variables. By integrating complementary data types, we avoid the need for structural assumptions. We illustrate theoretical and practical significance of the model and develop a reversible-jump MCMC learning procedure for learning modules and model parameters. We demonstrate the method accuracy in predicting modular structures from synthetic data and capability to learn regulatory modules in the *Mycobacterium tuberculosis* gene regulatory network.

1. Introduction

There is considerable interest in modeling dependency structures in a variety of applications. Examples include reconstructing regulatory relationships from gene expression data in gene networks or identifying influence structures from activity patterns such as purchases, posts, tweets, etc in social networks. Common approaches for learning dependencies include using Bayesian networks and factor analysis (Koller & Friedman, 2009).

Module networks (Segal et al., 2005; 2003) have been widely used to find structures (e.g. gene regulation) between groups of nodes (e.g. genes) denoted as modules, based on measurements of node-specific variables in a network (e.g. gene expression). The motivation lies in that nodes that are influenced or regulated by the same parent node(s), have the same conditional probabilities for their variables. For example, in gene regulatory networks, groups of genes respond in concert under certain environmental conditions (Qi & Ge, 2006) and are thus likely to be regulated by the same mechanism. In other domains, such as social networks, communities with similar interests or affiliations may have similar activity in posting messages (e.g. in twitter) in response to news-outbreaks or similar purchases in response to marketing advertisements (Kozinets, 1999; Aral et al., 2009).

However, inferring dependencies merely from node-specific variables can lead to higher rate of false positives (Michoel et al., 2007). For example, a dependency might be inferred between two unrelated nodes due to existing confounding variables. This can introduce arbitrary or too many parents for a module. To avoid over-fitting in inferring module networks, additional structural assumptions such as setting the maximum number of modules or maximum number of parents per module may be required. This in turn presents additional inductive bias and results become sensitive to assumptions. Moreover, searching through the entire set of candidate parents for each module is computationally infeasible.

Alternatively, we can take advantage of existing network data and by integrating node interactions with node variables, we can avoid structural assumptions. For exam-

Proceedings of the 31^{st} International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32.

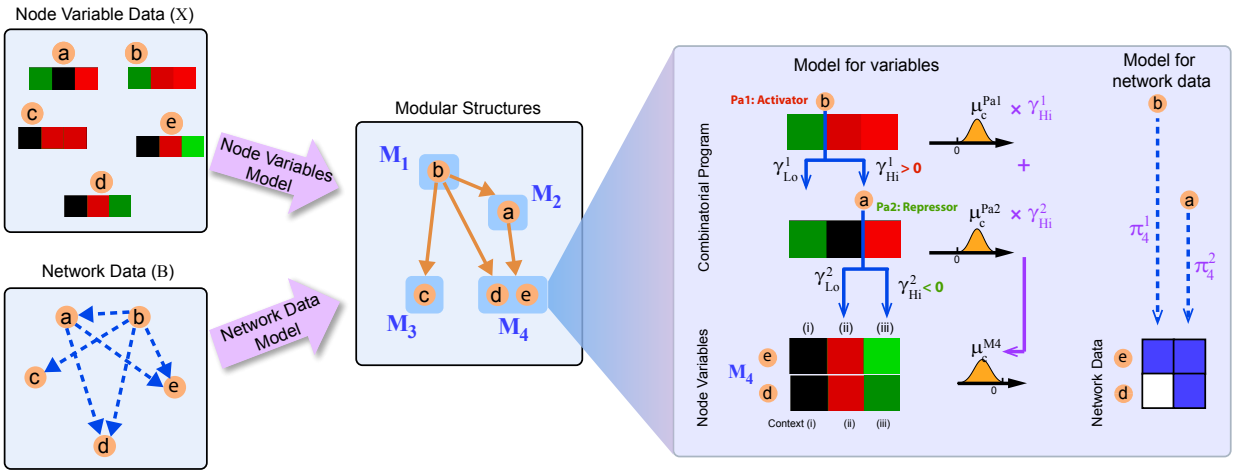

Figure 1. **Illustration of proposed model:** Modular structures are learned from node variables (e.g. gene expression) and network data (e.g. protein-DNA interactions). Node variables are color-coded ranging from green (low) to red (high). A number of parents are assigned to each module (orange links). A combinatorial program is inferred for each module; example shown for module M_4.

ple, to learn gene regulatory networks, we can use protein-DNA interaction data, which shows physical interactions between proteins of genes (known as Transcription Factors) with promoter regions of other genes, leading to regulation of transcription (and expression) of the latter genes. This data can be measured using chromatin immunoprecipitation of DNA-bound proteins, i.e. ChIP-ChIP or ChIP-Seq technologies, which have shown to be informative of regulation (Galagan et al., 2013; Liu et al., 2013; Celniker et al., 2009). As another example, to learn influence structures in a twitter network, we can integrate the network of who-follows-who with measurements of users activities.

Identifying modules or block structures from network data has been well-studied, e.g., using stochastic blockmodels (Wang & Wong, 1987; Snijders & Nowicki, 1997; Airoldi et al., 2008; 2013a) in the area of social network modeling (Goldenberg et al., 2009; Azari Soufiani & Airoldi, 2012; Choi et al., 2012). Stochastic blockmodels assume that nodes of a network are members of latent blocks, and describe their interactions with other nodes with a parametric model. However, models for inferring modular structures from both node variables and network data are relatively unexplored and of interest in many applications.

1.1. Contributions

In this paper, we propose an integrated probabilistic model inspired by module networks and stochastic blockmodels, to learn dependency structures from the combination of network data and node variables data. We consider network data in terms of directed edges (interactions) and model network data using stochastic blockmodels. Intuitively, by incorporating complementary data types, a node which is likely to have directed edges to members of a module as well as correlation with variables of module will be assigned as parent. A shorter version of this work was presented in (Azizi, 2013). The use of network data enhances computational tractability and scalability of the method by restricting the space of possible dependency structures. We also show theoretically that the integration of network data leads to model identifiability, whereas node variables alone can not, without extra structural assumptions.

Our model captures two types of relationships between variables of modules and their parents, including small changes of variables due to global dependency structure and condition-specific large effects on variables based on parent activities in each condition.

For estimation of parameters, we use a Gibbs sampler instead of the deterministic algorithm employed by Segal et al. to overcome some of the problems regarding multi-modality of model likelihood (Joshi et al., 2009). We also solve the problem of sensitivity to choice of maximum number of modules using a reversible-jump MCMC method which infers the number of modules and parents based on data. The probabilistic framework infers posterior distributions of assignments of nodes to modules and thus does not face restrictions of non-overlapping modules (Airoldi et al., 2008; 2013b).

1.2. Related Work

Other works have also proposed integrating different data types, mostly as prior information, for improvement in learning structures (Werhli & Husmeier, 2007; Imoto et al., 2003; Mitra et al., 2013). It is more natural to consider additional data types also as observations from a model

of dependency structures. Our model thus considers both network edges and node variables as data observed from the same underlying structure, providing more flexibility. Moreover, we utilize data integration to identify structures between groups of nodes (modules) as opposed to individual nodes. Despite the similarity in the framework of our model to module networks, our model for variables has differences in relating modules to their parents, giving more accurate and interpretable dependencies. Also, the integration of network data is novel. Regarding the learning procedure, prior work has been done on improving module network inference by using a Gibbs sampling approach (Joshi et al., 2009). We take a step further and use a reversible-jump MCMC procedure to learn the number of modules and parents from data as well as parameter posteriors. Our method can also allow restricting the number of modules based on context, with a narrow prior. By adjusting this prior, we have multi-resolution module detection.

2. Model of Modular Structures

In the framework of module networks, dependencies are learned from profiles of node variables (e.g. gene expressions) for each node (e.g. gene), as random variables $\{X_1, ..., X_N\}$. The idea is that a group of nodes with common parents (e.g. co-regulated genes) are represented as a module and have similar probability distributions for their variables conditioned on their shared parents (regulators). Figure 1 shows a toy example where node variable data are shown in green-to-red heatmaps and network data with dashed arrows (Airoldi, 2007). A module assignment function $\mathcal{A}$ maps nodes $\{1, ..., N\}$ to K non-overlapping modules. A dependency structure function $\mathcal{S}$ assigns a set of parents Pa_j from $\{1, ..., R\}$ known candidate parents (possible regulators/influencers), which are a subset of the N nodes, to module M_j (figure 1). In the toy example, nodes d, e are assigned to the same module M_4 and b, a are assigned as their parents. In cases where multiple parents drive a module, e.g. a, b affecting M_4, combinatorial effects are represented as a decision tree (regulatory program) and each combination of parents activities, defined as a context, is assigned to a cluster of conditions (experiments). In figure 1, parent b has an activating effect while a represses M_4, hence, e, d are active in context (ii) where only b is active and a is not. Inferring this decision tree in the context of different applications shows how multiple parents act together in influencing a group of nodes, e.g. in a gene network, multiple transcription-factors (TFs) act together to express a group of genes.

Given this framework, our model considers variables and network data as two types of observation from the same underlying modular structure. This structure is encoded based on assignments to modules ($\mathcal{A}$) and parents for each module ($\mathcal{S}$). In the example of gene networks, in each module, TF-gene interactions are likely to be observed between TFs and upstream regions of genes in the module while combinations of expressions of TFs explain expressions of genes.

2.1. Modeling Node Variables

We model variables for nodes $\{1, ..., N\}$ in each condition or sample $c \in 1, ..., C$ with a multivariate normal represented as $\mathbf{X_c} \sim \mathcal{N}(\boldsymbol{\mu}_c, \Sigma)$, where $\mathbf{X_c}$ is a $N \times 1$ vector, with N being the total number of nodes. The covariance and mean capture two different aspects of the model regarding global dependency structures and context-specific effects of parents, respectively, as described below.

We define the covariance $\boldsymbol{\Sigma}$ to be independent of conditions and representing the strength of potential effects of one variable upon another, if the former is assigned as a parent of the module containing the latter. In the example of gene expressions, Σ may represent the affinity of a Transcription-Factor protein to a target gene promoter. The modular dependencies between variables imposes a structure on $\boldsymbol{\Sigma}$. To construct this structure, we relate node variables to their parents through a regression $\mathbf{X_c} = W\mathbf{X_c} + \boldsymbol{\epsilon}$ where $\boldsymbol{\epsilon} = \mathcal{N}(\mathbf{m_c}, I)$. W is a $N \times N$ sparse matrix in which element W_{nr} is nonzero if variable r is assigned as a parent of the module containing variable n. Here we assume W_{nr} has the same value for $\forall n \in M_k, \forall r \in Pa_k$, which leads to identifiability of model (as explained in section 3. Then, assuming $I - W$ is invertible, $\mathbf{X_c} = (I - W)^{-1}\boldsymbol{\epsilon}$ which implies $\boldsymbol{\Sigma} = (I - W)^{-T}(I - W)^{-1}$. Therefore, we impose the modular dependency structure over Σ through W, which is easier to interpret based on $\mathcal{A}, \mathcal{S}$ assignments.

We define variable means $\boldsymbol{\mu}_c$, based on parents as described below. First, based on the modular structure of nodes, we can partition the mean vector as $\boldsymbol{\mu}_c = [\boldsymbol{\mu}_c^1 ... \boldsymbol{\mu}_c^K]^T$, where each $\boldsymbol{\mu}_c^k$ for $k = 1, ..., K$ is a $1 \times N_k$ vector with N_k equal to the number of nodes in module k. In modules where there is more than one parent assigned, combinations of different activities of parents, creating a context, can lead to different effects. The binary state of parent $r \in Pa_k$ is defined by comparing its mean to a split-point z_k^r, corresponding to a mixture coefficient for that state γ_{Lo}^r or γ_{Hi}^r, as: $\gamma_c^r = \gamma_{Lo}^r H(z_k^r - \mu_c^r) + \gamma_{Hi}^r H(\mu_c^r - z_k^r)$, where $H(\cdot)$ is a unit step function.

The combination of different activities are represented as a decision tree for each module k (figure 1). We represent a context-specific program as dependencies of variable means on parents activities in each context, such that $\boldsymbol{\mu}_c^k$ for module k is a linear mixture of means for parents of that module: $\boldsymbol{\mu}_c^k = \sum_{r=1}^{R_k} \boldsymbol{\gamma}_c^r \boldsymbol{\mu}_c^{Pa_k^r}$ where R_k is the number of parents Pa_k and $\boldsymbol{\gamma}_c^r$ are similar for all conditions c occurring in the same context. Thus, in general we can

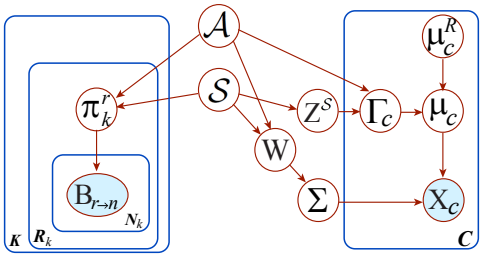

Figure 2. **Graphical representation of model:** The assignments of nodes to modules $\mathcal{A}$ and parents for modules $\mathcal{S}$ represent modular dependency structures, from which we observe node variables $\mathbf{X_c}$ in each condition c and network data $B_{r\to n}$ between a parent r and a node n. Means of node variables μ_c are determined from parent means μ_c^R with mixing coefficients Γ determined based on parent split-points Z.

write $\boldsymbol{\mu}_c = \Gamma_c \boldsymbol{\mu}_c^R$, where $\boldsymbol{\mu}_c^R$ contains the means of parents $1, ..., R$ in condition c. The $N \times R$ matrix Γ_c has identical rows for all variables in one module based on the assignment functions $\mathcal{A}, \mathcal{S}$. The graphical model is summarized in figure 2. Thus the model for object variables would be: $\mathbf{X_c} \sim \mathcal{N}(\Gamma_c \boldsymbol{\mu}_c^R, (I-W)^{-T}(I-W)^{-1})$.

Given independent conditions, the probability of data $\mathbf{X} = [\mathbf{X_1}, ..., \mathbf{X_C}]$ for C conditions given parameters can be written as multiplication of multivariate normal distributions for each condition: $P(\mathbf{X}|\mathcal{A}, \mathcal{S}, \Theta, \Sigma, Z^S) = \prod_{c=1}^{C} P(\mathbf{X_c}|\mathcal{A}, \mathcal{S}, \theta_c, \Sigma, Z^S)$, where $\Theta = \{\theta_1, ..., \theta_C\}$ denotes the set of condition-specific parameters $\theta_c = \{\boldsymbol{\mu}_c^R, \Gamma_c\}$ for $c = 1, ..., C$ and Z^S denotes the set of parent split-points for all modules. Then for each condition we have: $P(\mathbf{X_c}|\mathcal{A}, \mathcal{S}, \theta_c, \Sigma, Z^S) = \frac{1}{(2\pi)^{N/2}|\boldsymbol{\Sigma}^{1/2}|} exp(-\frac{1}{2}(\mathbf{X_c} - \boldsymbol{\mu}_c)^T \boldsymbol{\Sigma}^{-1}(\mathbf{X_c} - \boldsymbol{\mu}_c))$.

Hence, this model provides interpretations for two types of influences of parents. By relating the distribution mean for variables in each module and in each condition to means of their assigned parents (figure 1.B), we model condition-specific effects of parents. Based on the states of parents in different contexts (partitions of conditions), this leads to a bias or large signal variations in node variables. Whereas, small signal changes (linear term) are modeled through the covariance matrix Σ which is independent of condition and is only affected by the global wiring imposed by dependency structures.

2.2. Modeling Network Data

Network data, as a directed edge between a parent $r \in \{1, ..., R\}$ and node $n \in M_k$, when r is assigned as a parent of the module $r \in Pa_k$ is defined as a directed link $B_{r\to n}$ where

$$P(B_{r\in Pa_k \to n\in M_k}|\mathcal{A}, \mathcal{S}, \pi_k^r) \sim Bernoulli(\pi_k^r) \quad (1)$$

The parameter π_k^r defines the probability of parent r influencing module M_k (figure 2). In the gene network example, an interaction between a Transcription Factor protein binding to a motif sequence, upstream of target genes, which is common in all genes of a module can be observed using ChIP data. Therefore, directed interactions from parents to all nodes in a module would be $P(B_{M_k}|\mathcal{A}, \mathcal{S}, \boldsymbol{\pi}_k) = \prod_{r\in Pa_k} \prod_{n\in M_k} P(B_{r\to n}|\mathcal{A}, \mathcal{S}, \pi_k^r)$, where $\boldsymbol{\pi_k}$ is the vector of π_k^r for all $r \in Pa_k$ and for all nodes we have:

$$P(\mathbf{B}|\mathcal{A}, S, \boldsymbol{\pi}) = \prod_{k=1}^{K} \prod_{r\in Pa_k} \prod_{n\in M_k} P(B_{r\to n}|\mathcal{A}, \mathcal{S}, \pi_k^r)$$
$$= \prod_{k=1}^{K} \prod_{r\in Pa_k} (\pi_k^r)^{s_{rk}} (1-\pi_k^r)^{|M_k|-s_{rk}}$$
$$\prod_{r'\notin Pa_k} (\pi_0)^{s_{r'k}} (1-\pi_0)^{|M_k|-s_{r'k}} \quad (2)$$

with $\boldsymbol{\pi} = \{\boldsymbol{\pi}_1, ..., \boldsymbol{\pi}_K\}$ and $s_{rk} = \sum_{n\in M_k}(B_{r\to n})$ is the sufficient statistic for the network data model and $|M_k|$ is the number of nodes in module k and π_0 is the probability that any non-parent can have interaction with a module. In gene regulatory networks, π_0 can be interpreted as basal level of physical binding that may not necessarily affect gene transcription and thus regulate a gene.

In the context of stochastic blockmodels, the group of parents assigned to each module can be considered as an individual block and thus our model can represented as overlapping blocks of nodes.

The likelihood of the model $\mathcal{M} = \{\mathcal{A}, \mathcal{S}, \Theta, \Sigma, Z^S, \boldsymbol{\pi}\}$ given the integration of node variables and network data is: $P(\mathbf{X}, \mathbf{B}|\mathcal{M}) = P(\mathbf{X}|\mathcal{A}, \mathcal{S}, \Theta, \Sigma, Z^S) P(\mathbf{B}|\mathcal{A}, \mathcal{S}, \boldsymbol{\pi})$. With priors for parameters $\mathcal{M}$ the posterior likelihood is: $P(\mathcal{M}|\mathbf{X}, \mathbf{B}) \propto P(\mathcal{M}) P(\mathbf{X}, \mathbf{B}|\mathcal{M})$.

3. Theory: Model Identifiability

Our method uses network data to avoid extra structural assumptions. In this section we formalize this idea through the identifiability of the proposed model. This property is important for interpretability of learned modules. Module networks and generally multivariate normal models for object variables can be un-identifiable, and imposing extra structural assumptions is necessary to overcome this. Here, we illustrate that the integrated learning proposed in this paper resolves the un-identifiability issue. First, we show that modeling node variables alone is identifiable only under very specific conditions. Then, we will restate some results from (Latouche et al., 2011) on the identifiability of overlapping block models. Using this result we show the identifiability of the model under some reasonable conditions.

Lemma 1. Node Variables Model: *For the model of node-specific variables* $\mathbf{X}$*, if we have:* $P(\mathbf{X}|\{\mathcal{A},\mathcal{S}\}',\Theta',\Sigma') = P(\mathbf{X}|\{\mathcal{A},\mathcal{S}\},\Theta,\Sigma)$

1. *Then, we can conclude:* $\mu' = \mu$ *and* $\Sigma' = \Sigma$.
2. *If we further assume* $\{\mathcal{A},\mathcal{S}\} = \{\mathcal{A},\mathcal{S}\}'$ *and that each module has at least two non parent nodes and* $\sum_k |Pa_k| < N$ *and the covariance matrix* Σ *is invertible, we can conclude:* $\Theta = \Theta'$, $W = W'$.

Proof presented in (Azizi et al., 2014).

The above lemma provides identifiability for the case where the structure $\{\mathcal{A},\mathcal{S}\}$ is assumed to be known. However, in the case where we don't have the structure, the parameterizations of multivariate normal (μ and Σ) can be written in multiple ways in terms of Θ and $\{\mathcal{A},\mathcal{S}\}$. This is due to existence of multiple decompositions for the covariance matrix. In the following, we will use a theorem for identifiability of overlapping block models from (Latouche et al., 2011) which is an extension of the results in (Allman et al., 2009). The results provide conditions for overlapping stochastic block models to be identifiable.

Theorem 1. Network Data Model: *If we have* $P(\mathbf{B}|\{\mathcal{A},S\},\boldsymbol{\pi}) = P(\mathbf{B}|\{\mathcal{A},S\}',\boldsymbol{\pi}')$*, then:* $\{\mathcal{A},S\} = \{\mathcal{A},S\}'$ *with a permutation and* $\boldsymbol{\pi} = \boldsymbol{\pi}'$ *(except in a set of parameters which have a null Lebesgue measure) (Proof direct result of Theorem 4.1 in Latouche et al. (2011) as described in Azizi et al. (2014)).*

Using the above Theorem and Lemma 1 we can have the following Theorem for the identifiability of the model.

Theorem 2. Identifiability of Model: *If we have:* $P(\mathbf{B}|\{\mathcal{A},S\},\boldsymbol{\pi}) = P(\mathbf{B}|\{\mathcal{A},S\}',\boldsymbol{\pi}')$ *and* $P(\mathbf{X}|\{\mathcal{A},\mathcal{S}\}',\Theta',\Sigma') = P(\mathbf{X}|\{\mathcal{A},\mathcal{S}\},\Theta,\Sigma)$ *with assuming that each module has at least two non-parent nodes and* $\sum_k |Pa_k| < N$ *and the covariance matrix* Σ *is invertible, then:* $\{\mathcal{A},S\} = \{\mathcal{A},S\}'$ *with a permutation,* $\boldsymbol{\pi} = \boldsymbol{\pi}'$, $\Theta = \Theta'$ *and* $W = W'$ *(except in a set of parameters which have a null Lebesgue measure) (Proof in Azizi et al. (2014)).*

This Theorem states the theoretical effect of integrated modeling on identifiability of modular structures, given that the sum of number of parents is less than the number of nodes (as is common in gene regulatory networks).

4. Parameter Estimation using RJMCMC

We use a Gibbs sampler to obtain the posterior distribution $P(\mathcal{M}|\mathbf{X},\mathbf{B})$ and design Metropolis-Hastings samplers for each of the parameters $\Theta,\Sigma,\boldsymbol{\pi}$ conditioned on the other parameters and data $\mathbf{X},\mathbf{B}$. We use Reversible-Jump MCMC (Green, 1995) for sampling from conditional distributions of the assignment and structure parameters $\mathcal{A},\mathcal{S}$.

4.1. Learning Parameters $\Theta,\Sigma,Z^S,\boldsymbol{\pi}$.

To update the means, we only need to sample one value for means of parents assigned to the same module. This set of means of distinct parents $\boldsymbol{\mu}_c^{\mathbf{R}}$ are sampled with a Normal proposal (Algorithm 1). Similarly we sample the parameters $\boldsymbol{\gamma}_c^r$, z_k^r and π_k^r, corresponding to parent $r \in Pa_k$ of module k, from normal distributions. To update covariance Σ, each distinct element of the regression matrix W corresponding to a module k, denoted as w_k, is updated. Due to the symmetric proposal distribution, the proposal is accepted with probability $P_{mh} = \min\{1, \frac{P(\mathcal{M}^{(j+1)}|X,B)}{P(\mathcal{M}^{(j)}|X,B)}\}$. The conditions required for identifiability (from Theorem 1) are enforced in each iteration.

where $\mathcal{M}^{(j)} = \{\mathcal{A},\mathcal{S},\Theta,\Sigma,Z^S\boldsymbol{\pi}\}^{(j)}$.

Algorithm 1 RJMCMC for sampling parameters

Inputs:
Node Variables Data $\mathbf{X}$
Network Data $\mathbf{B}$
for iterations $j = 1$ **to** J **do**
  Sample $\mathcal{A}^{(j+1)}$ given $\mathcal{A}^{(j)}$ using Alg 2 in (Azizi et al., 2014)
  Sample $\mathcal{S}^{(j+1)}$ given $\mathcal{S}^{(j)}$ using Alg 3 in (Azizi et al., 2014)
  for modules $k = 1$ **to** $K^{(j)}$ **do**
    Propose $w_k^{(j+1)} \sim \mathcal{N}(w_k^{(j)}, I)$
    Accept with probability P_{mh}; update $\Sigma^{(j+1)}$
    for parents $r = 1$ **to** R_k **do**
      Propose $z_k^{r(j+1)} \sim \mathcal{N}(z_k^{r(j)}, I)$; accept with P_{mh}
      Propose $\pi_k^{r(j+1)} \sim \mathcal{N}(\pi_k^{r(j)}, I)$; accept with P_{mh}
    end for
  end for
  for condition $c = 1$ **to** C **do**
    Propose $\boldsymbol{\mu}_c^{\mathbf{R}(j+1)} \sim \mathcal{N}(\boldsymbol{\mu}_c^{\mathbf{R}(j)}, I)$; accept with P_{mh}
    Propose $\boldsymbol{\gamma}_c^{\mathbf{R}(j+1)} \sim \mathcal{N}(\boldsymbol{\gamma}_c^{\mathbf{R}(j)}, I)$; accept with P_{mh}
  end for
end for

4.2. Learning assignments $\mathcal{A},\mathcal{S}$.

Learning the assignment of each node to a module, involves learning the number of modules. Changing the number of modules however, changes dimensions of the parameter space and therefore, densities will not be comparable. Thus, to sample from $P(\mathcal{A}|\mathcal{S},\Theta,\Sigma,,Z^S\boldsymbol{\pi},\mathbf{X},\mathbf{B})$, we use the Reversible-Jump MCMC method (Green, 1995), an extension of the Metropolis-Hastings algorithm that allows moves between models with different dimensionality. In each proposal, we consider three close move schemes of increasing or decreasing the number of modules by one, or

not changing the total number. For increasing the number of modules, a random node is moved to a new module of its own and for decreasing the number, two modules are merged. In the third case, a node is randomly moved from one module to another module, to sample its assignment (Algorithm 2 in (Azizi et al., 2014)).

To sample from the dependency structure (assignment of parents) $P(\mathcal{S}|\mathcal{A}, \Theta, \Sigma, Z^S\boldsymbol{\pi}, \mathbf{X}, \mathbf{B})$, we also implement a Reversible-Jump method, as the number of parents for each module needs to be determined. Two proposal moves are considered for $\mathcal{S}$ which include increasing or decreasing the number of parents for each module, by one (Algorithm 3 in (Azizi et al., 2014)).

5. Results

5.1. Synthetic Data

We first tested our method on synthetic node-variables and network data generated from the proposed model. A dataset was generated for $N = 200$ nodes in $K = 4$ modules with $C = 50$ conditions for each node variable. Parents were assigned from a total of $R = 10$ number of candidates. Parameters π, γ and W were chosen randomly, preserving parameter sharing of modules. The inference procedure was run for 20,000 samples. Exponential prior distributions were used for number of parents assigned to each module, to avoid over-fitting. Figure 3 shows the autocorrelation for samples of variable mean μ_c^n for an example gene. The samples become independent after a lag and thus we removed the first $10,000$ iterations as burn-in period. Samples from posteriors, including the number of modules K, exhibit standard MCMC movements around the actual value (actual $K = 4$). We also calculated the true positive rate and false positive rates based on actual dependency links. We repeated the estimation of true positive and false positive rates for 100 random datasets with the same size as mentioned and computed the average ROC for the model (figure 3). As comparison, for each generated dataset, we also tested the sub-model for variable data (excluding the model for network data) to infer links. We performed bootstrapping on sub-samples with size 1000 to compute variance of AUC (area under curve) and paired t-tests confirmed improved performance of integrated model compared to the variables sub-model ($p < 0.05$).

The parameter sharing property in modular structures allows parallel sampling of parameters w_k and $\gamma^r_{(k)}$, z^r_k,π^r_k for each module k, in each iteration and in different conditions. We used Matlab-MPI for this implementation. It takes an average of 36 ± 8 seconds to generate 100 samples for $N = 200$, $C = 50$, $R = 10$ on an i5 3.30GHz Intel(R). For further enhancement, module assignments were initialized by k-means clustering of variables.

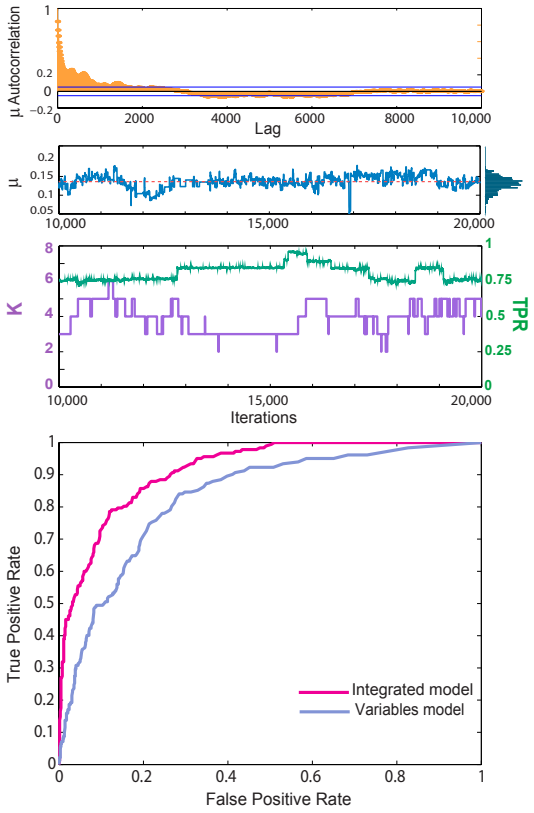

Figure 3. **Results sythetic data:** Autocorrelation for an example variable mean (top); gibbs samples and posterior after burn-in period (actual mean shown with red line); number of modules (purple) and true positive rate of recovered links (green), ROC curve for integrated model and variables model (bottom)

5.2. M. tuberculosis Gene Regulatory Network

We applied our method to identify modular structures in the Mycobacterium tuberculosis (MTB) regulatory network. MTB is the causative agent of tuberculosis disease in humans and the mechanisms underlying its ability to persist inside the host are only partially known (Flynn & Chan, 2001). We used interaction data identified with ChIP-Seq of 50 MTB transcription factors and expression data for different induction levels of the same factors in 87 experiments, from a recent study by (Galagan et al., 2013). Only bindings of factors to upstream intergenic regions were considered. We tested our method on 3072 MTB genes which had binding from at least one of these factors and performed 100,000 iterations on the combination of the two datasets. For each gene, we inferred the mode of its assignments to modules (after removing burn-in samples) and obtained 29 modules in total. The largest modules and the assigned regulators are shown in figure 4. The identified modules are enriched for functional annotations of genes (Azizi et al., 2014).

For each module, the number of assigned genes and ex-

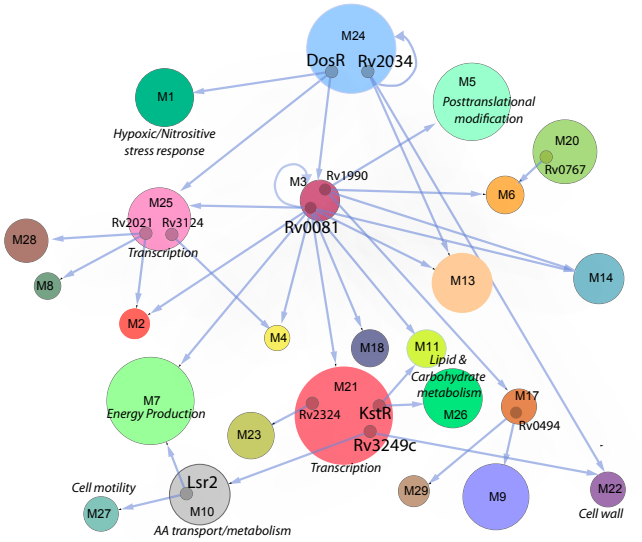

Figure 4. **Regulatory structures between largest modules inferred for MTB**: Regulators assigned to each module are shown; the size of circles are proportional to number of genes assigned to the module. Enriched functional annotations are highlighted; details are in Azizi et al. (2014).

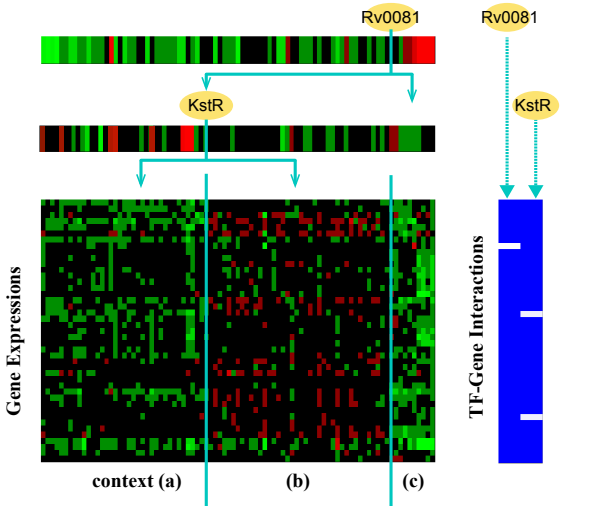

Figure 5. Inferred regulatory program for module M11 of fig. 4 showing that either of Rv0081 and KstR can repress the module in contexts (a) and (c)

amples of previously studied genes are presented. The identified regulators of each module and enriched annotations confirm known functions for some regulators, such as the role of KstR (Rv3574) in regulating lipid metabolism (Kendall et al., 2007), confirmed in modules M26 and M11; and DosR (Rv3133c) in nitrosative stress response (Voskuil et al., 2003) (module M1) and transcription (Rustad et al., 2008) (module M25). Novel functions for other regulators and the combinations of regulators acting together are also presented.

As shown in figure 4, many modules are controlled by more than one regulator, highlighting the significance of combinatorial regulations (see supplemantary material for interpretations). One inferred module is M11 shown in figure 5 which is regulated by Rv0081 and KstR (Rv3574). Rv0081 is known to be involved in hypoxic adaptation (Galagan et al., 2013) while KstR is known to be involved in cholesterol and lipid catabolism (Kendall et al., 2007) and the module is enriched for "Energy production and conversion" and "Lipid transport and metabolism" COG categories (table 1 in Azizi et al. (2014)). The inferred program in figure 5 shows that either of the two regulators can repress the expression of the 48 genes assigned to this module, which include lipases and genes involved in fatty acid β-oxidation and triacylglycerides cycle metabolic pathways. KstR itself is also regulated by Rv0081, and thus Rv0081 regulates lipid metabolism genes through KstR. Figure 3 in supplemantary material shows another module M25 containing 161 genes, with two hypoxic adaptation regulators mediating the induction of a second hierarchy of regulators with a time delay, explaining a late hypoxic response.

We showed in section 3 that integration of network data has theoretical advantages in terms of model identifiability. Here, we show that it can also reduce the number of false positive regulatory links in MTB data. As a gold standard, we used previously validated links (by EMSA, RTq-PCR) for two MTB regulators, including 48 known links for DosR from (Voskuil et al., 2003) and 72 known links for KstR from (Kendall et al., 2007). We calculated the area under precision-recall for our method by comparing posterior probabilities for DosR and KstR links to known links (table 1). As comparison, we also applied common methods shown to have best performance in DREAM challenge contests (Marbach et al., 2012) in inferring regulatory networks from gene expression only. These include Mutual Information between expression profiles (MI), CLR (Faith et al., 2007), GENIE3 (Irrthum et al., 2010). We applied these on the above MTB expression data, and compared the inferred links to the gold standard set. As the number of validated links in MTB are small, we also scored the predictions from co-expression methods to the MTB ChIP-Seq data (Galagan et al., 2013) for the same two regulators. Also, none of these methods assume modular structures.

We then applied Module Networks (Segal et al., 2005) to the same expression dataset and compared predictions to known links and ChIP-Seq data (table 2). We set the maxmimum number of modules to 10 and constrained the candidate pool of regulators to the 50 ChIPped regulators only. On average 2.8 ± 0.63 regulators were assigned to each module, with a mode of 3, whereas the ChIP-Seq network shows a mode of 1 for in-degree of genes (Galagan et al., 2013), i.e. most genes have only one regulator binding. As the predicted links from module networks are de-

Table 1. Area under precision-recall AUPR(%) calculated for link prediction using proposed method and other common co-expression methods, applied to MTB data. The predictions are scored vs known and ChIP-Seq links for two regulators

Gold Standard	Validated Links		ChIP-Seq Links	
Regulator	DosR	KstR	DosR	KstR
No. of Targets	(48)	(72)	(528)	(503)
MI	39.04	9.24	25.00	17.85
CLR	48.25	9.37	21.44	16.77
GENIE3	**62.26**	31.37	21.55	19.44
Proposed Model	**72.13**	**65.72**	**79.62**	**70.06**

terministic, an AUPR score can not be reported, thus we compared to precision and recall of posterior mode from our models. Note small precision values are due to small number of validated links, i.e. if a link is not validated experimentally it may not be wrong. For a fair comparison of models without the effect of interaction data, we also compared to performance of our model for variables data only (table 2). These results show that module networks and in general co-expression methods have many false positives and integrating interaction data is necessary for inference of direct regulatory relationships.

Table 2. Percentage of Precision (P) and Recall (R) for link prediction using module networks and proposed models.

Gold Standard	Validated Links				ChIP-Seq Links			
Regulator	DosR		KstR		DosR		KstR	
	P	R	P	R	P	R	P	R
Module Networks	3.8	81.2	6.5	86.1	40.1	76.3	35.8	67.4
Proposed Model for Variables (mode)	4.6	77.1	7.2	77.8	55.0	83.7	52.5	80.5
Proposed Integrated Model (mode)	6.5	89.6	10.6	84.7	75.4	93.4	83.6	95.6

6. Conclusion

We proposed a model for learning dependency structures between modules, from network data and node variables data. We showed that the assumption of shared parents and parameters for nodes in a module, together with integration of network data deals with under-determination and un-identifiability, improves statistical robustness and avoids over-fitting. We presented a reversible-jump inference procedure for learning model posterior. Our results showed high performance on synthetic data and interpretable structures on synthetic data and real data from *M. tuberculosis* gene network. Results for MTB gene regulatory network revealed feed-forward loops and insights into condition-specific regulatory programs for lipid metabolism and hypoxic adaptation. One future direction is to propose faster algorithms based on generalized method of moments (Azari Soufiani et al., 2014; 2013; Anandkumar et al., 2012) for estimators of this model.

Acknowledgments

We acknowledge funding from the Hariri Institute for Computing and Computational Science & Engineering, the National Institute of Health under grants HHSN272200800059C and R01 GM096193, the National Science Foundation under grant IIS-1149662, the Army Research Office under grant MURI W911NF-11-1-0036, and from an Alfred P. Sloan Research Fellowship.

References

Airoldi, E.M. Getting started in probabilistic graphical models. *PLoS Computational Biology*, 3(12):e252, 2007.

Airoldi, E.M., Blei, D.M., Fienberg, S.E., and Xing, E.P. Mixed membership stochastic blockmodels. *The Journal of Machine Learning Research*, 9:1981–2014, 2008.

Airoldi, E.M., Costa, T.B., and Chan, S.H. Stochastic blockmodel approximation of a graphon: Theory and consistent estimation. In *Advances in Neural Information Processing Systems (NIPS)*, volume 26, pp. 692–700, 2013a.

Airoldi, E.M., Wang, X., and Lin, X. Multi-way blockmodels for analyzing coordinated high-dimensional responses. *Annals of Applied Statistics*, 7(4):2431–2457, 2013b.

Allman, E.S., Matias, C., and Rhodes, J.A. Identifiability of parameters in latent structure models with many observed variables. *The Annals of Statistics*, 37(6A):3099–3132, 2009.

Anandkumar, A., Hsu, D., and Kakade, S.M. A method of moments for mixture models and hidden markov models. *arXiv:1203.0683*, 2012.

Aral, S., Muchnik, L., and Sundararajan, A. Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks. *Proceedings of the National Academy of Sciences*, 106(51):21544–21549, 2009.

Azari Soufiani, H. and Airoldi, E.M. Graphlet decomposition of a weighted network. *Journal of Machine Learning Research*, (W&CP 22 (AISTATS)):54–63, 2012.

Azari Soufiani, H., Chen, W., Parkes, D.C., and Xia, L. Generalized method-of-moments for rank aggregation. In *Advances in Neural Information Processing Systems*, pp. 2706–2714, 2013.

Azari Soufiani, H., Parkes, D.C., and Xia, L. Computing parametric ranking models via rank-breaking. In *Proceedings of The 31st International Conference on Machine Learning*, pp. 360–368, 2014.

Azizi, E. Joint learning of modular structures from multiple data types. In *NIPS Workshop of Frontiers of Network Analysis: Methods, Models, and Applications*, 2013.

Azizi, E., Airoldi, E.M., and Galagan, J.E. Learning modular structures from network data and node variables. *arXiv:1405.2566*, 2014.

Celniker, S.E., Dillon, L., Gerstein, M.B., Gunsalus, K.C., Henikoff, S., Karpen, G.H., Kellis, M., Lai, E.C., Lieb, J.D., MacAlpine, D.M., et al. Unlocking the secrets of the genome. *Nature*, 459(7249):927–930, 2009.

Choi, D.S., Wolfe, P.J., and Airoldi, E.M. Stochastic blockmodels with a growing number of classes. *Biometrika*, 99(2):273–284, Jun. 2012.

Faith, J.J., Hayete, B., Thaden, J.T., Mogno, I., Wierzbowski, J., et al. Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles. *PLoS biology*, 5 (1):e8, 2007.

Flynn, J.L. and Chan, J. Tuberculosis: latency and reactivation. *Infection and immunity*, 69(7):4195–4201, 2001.

Galagan, J.E., Minch, K., Peterson, M., Lyubetskaya, A., Azizi, E., et al. The mycobacterium tuberculosis regulatory network and hypoxia. *Nature*, 499(7457):178–183, 2013.

Goldenberg, A., Zheng, A. X., Fienberg, S. E., and Airoldi, E. M. A survey of statistical network models. *Foundations and Trends in Machine Learning*, 2(2):129–233, Feb. 2009.

Green, P.J. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

Imoto, S., Higuchi, T., Goto, T., Tashiro, K., Kuhara, S., and Miyano, S. Combining microarrays and biological knowledge for estimating gene networks via bayesian networks. *Proc. Computational Systems Bioinformatics*, 2003.

Irrthum, A., Wehenkel, L., Geurts, P., et al. Inferring regulatory networks from expression data using tree-based methods. *PLoS One*, 5(9):e12776, 2010.

Joshi, A., De Smet, R., Marchal, K., Van de Peer, Y., and Michoel, T. Module networks revisited: computational assessment and prioritization of model predictions. *Bioinformatics*, 25(4):490–496, 2009.

Kendall, S.L., Withers, M., Soffair, C.N., Moreland, N.J., Gurcha, S., et al. A highly conserved transcriptional repressor controls a large regulon involved in lipid degradation in mycobacterium smegmatis and mycobacterium tuberculosis. *Molecular microbiology*, 65(3):684–699, 2007.

Koller, D. and Friedman, N. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

Kozinets, R.V. E-tribalized marketing?: The strategic implications of virtual communities of consumption. *European Management Journal*, 17(3):252–264, 1999.

Latouche, P., Birmelé, E., and Ambroise, C. Overlapping stochastic block models with application to the french political blogosphere. *The Annals of Applied Statistics*, 5(1):309–336, 2011.

Liu, Y., Qiao, N., Zhu, S., Su, M., Sun, N., Boyd-Kirkup, J., and Han, J.-D. A novel bayesian network inference algorithm for integrative analysis of heterogeneous deep sequencing data. *Cell Research*, 23(3):440–443, 2013.

Marbach, D., Costello, J.C., Küffner, R., Vega, N.M., Prill, R.J., et al. Wisdom of crowds for robust gene network inference. *Nature methods*, 2012.

Michoel, T., Maere, S., Bonnet, E., Joshi, A.and Saeys, Y., et al. Validating module network learning algorithms using simulated data. *BMC Bioinformatics*, 8(Suppl 2):S5, 2007.

Mitra, K., Carvunis, A., Ramesh, S.K., and Ideker, T. Integrative approaches for finding modular structure in biological networks. *Nature Reviews Genetics*, 14(10):719–732, 2013.

Qi, Y. and Ge, H. Modularity and dynamics of cellular networks. *PLoS Computational Biology*, 2(12):e174, 2006.

Rustad, T.R., Harrell, M.I., Liao, R., and Sherman, D.R. The enduring hypoxic response of mycobacterium tuberculosis. *PLoS One*, 3(1):e1502, 2008.

Segal, E., Shapira, M., Regev, A., Pe'er, D., Botstein, D., Koller, D., and Friedman, N. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nature genetics*, 34(2):166–176, 2003.

Segal, E., Pe'er, D., Regev, A., Koller, D., and Friedman, N. Learning module networks. *Journal of Machine Learning Research*, (6):557–588, 2005.

Snijders, T.A.B. and Nowicki, K. Estimation and prediction for stochastic blockmodels for graphs with latent block structure. *Journal of Classification*, 14(1):75–100, 1997.

Voskuil, M.I., Schnappinger, D., Visconti, K.C., Harrell, M.I., Dolganov, G.M., Sherman, D.R., and Schoolnik, G.K. Inhibition of respiration by nitric oxide induces a mycobacterium tuberculosis dormancy program. *The Journal of experimental medicine*, 198(5):705–713, 2003.

Wang, Y.J. and Wong, G.Y. Stochastic blockmodels for directed graphs. *Journal of the American Statistical Association*, 82(397):8–19, 1987.

Werhli, A.V. and Husmeier, D. Reconstructing gene regulatory networks with bayesian networks by combining expression data with multiple sources of prior knowledge. *Statistical Applications in Genetics and Molecular Biology*, 6(1):15, 2007.