Joint Learning of Modular Structures from Multiple Data Types

Elham Azizi
Bioinformatics Program, Department of Biomedical Engineering
Boston University
Boston, MA 02215
elham@bu.edu

Abstract

A commonly used technique for understanding underlying dependency structures among objects is module networks by Segal et al., which assumes a shared conditional probability distribution for objects within one module. However, learning structures from object variables alone can lead to spurious dependencies and to avoid over-fitting, imposing structural assumptions may be required. We propose an extended model inspired by module networks and stochastic blockmodels for joint learning of structures from observed object variables (e.g. gene expression in gene regulatory networks) and relational data among objects (e.g. protein-DNA interactions). By integrating complementary data types, we avoid additional structural assumptions. We illustrate theoretical and practical significance of the model and developed a reversible-jump MCMC learning procedure for learning modules and model parameters. We demonstrate the accuracy and scalability of our method for synthetic and genomic datasets.

1 Introduction

There is considerable interest in modeling dependencies between a large number of objects based on observations in a variety of applications. Examples include reconstructing regulatory relationships from gene expression data in biological networks or identifying influence from purchasing patterns in social networks. Common approaches for learning dependencies include using Bayesian networks and factor analysis [13].It can be beneficial to identify groups or modules within these interaction networks. Modular behavior can be natural and interpretable in some domains such as gene regulatory networks, which consist of partitions of genes acting in concert under certain environmental cues [19].In other domains, e.g. in social networks, communities with similar interests or affiliations may have similar behavior in communicating messages in response to news-outbreaks or similar purchases in response to marketing advertisements [14]. Computational advantages of imposing a modular structure include parameter-sharing of objects in a module. This deals with under-determination (un-identifiability) of the problem in complex networks, improves statistical robustness and avoids over-fitting to individual variables, with the assumption of shared parents in the network.

The work of module networks [21, 22] has been widely used to find dependency structures (e.g. gene regulation) between groups of objects, denoted as modules, based on measurements of profiles of measured variables pertaining to objects (e.g. gene expression). However, inferring dependencies from object variables alone can lead to false positive predictions [17]. For example, a link might be inferred between two unrelated objects due to correlated behavior. To avoid over-fitting, additional structural assumptions such as maximum number of modules or maximum number of parents per module may be required for utilizing this method, which present additional inductive bias. A deterministic optimization algorithm is used by Segal et al. to search simultaneously for a partition of

objects into modules and a dependency structures for each module, which involves multiple local optima [12]. Furthermore, searching through the entire set of candidate parents for each module, introduces arbitrary selections among correlated parents or learns too many parents and solutions are sensitive to selection of maximum number of modules and parents.

Identifying modules or blocks in networks using interaction (relational) data has been well-studied in works of stochastic blockmodels [23, 1, 2] in the field of social network modeling [4]. In this paper, we propose an integrated probabilistic model inspired by module networks and stochastic blockmodels, to learn dependency structures from observations of individual objects and relational information between objects. In the biological application, we can integrate gene expression data with protein-DNA interaction data obtained from ChIP-ChIP or ChIP-Seq technologies, which have shown to be informative of regulation [9, 16]. Thus, we assign those Transcription Factors (TFs) that have both physical interaction with genes and predictive power in explaining their expression, as their regulators. Examples in social networks can include integrating number of posts on facebook (as object variables) with number of messages sent between friends (relational data) to identify structures of influence. Incorporating complementary relational information, if available, can improve accuracy by avoiding false assignments of indirect and correlated objects as parents [7]. Also, it enhances computational tractability and scalability of the method by restricting the space of possible dependency structures.

Our model captures two types of global and condition-specific relationships between observed variables for objects and their parents. For estimation of parameters, we use a Gibbs sampler instead of the optimization method employed by Segal et al. to overcome some of the problems regarding multi-modality of model likelihood. 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 regulators based on data. The probabilistic framework infers posterior distributions of assignments of genes to modules, and thus does not face restrictions of non-overlapping modules [2].

1.1 Related Literature

In terms of joint learning, other works have also proposed integrating different data types, mostly as prior information, for improvement in learning structures [24, 5], whereas our model considers relational interactions as data observed from the underlying structure. Also, here we utilize data integration to identify structures between modules (groups of objects). In terms of improving module networks, although the framework of our model is similar, our model for observed variables has differences in how object variables are related to their parents, giving more interpretable dependencies. Moreover, the integration of relational data is novel. Regarding the learning procedure, prior work has been done on improving module network inference by using a Gibbs sampling approach [12]. 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.

2 Model

In the framework of module networks, dependencies are learned from profiles of measured variables (e.g. gene expressions) for each object (e.g. gene), as random variables $\{X_1, ..., X_N\}$. The idea is that a group of objects 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). A module assignment function $\mathcal{A}$ maps objects $\{1, ..., N\}$ to K non-overlapping modules. A dependency structure function $\mathcal{S}$ assigns a set of parents Pa_j from $\{1, ..., R\}$ known candidate parents/regulators, which are a subset of the N objects, to module M_j (figure 1.A). In the case of multiple parents for a module, combinatorial interactions [6] can occur, represented as a regression tree in which clusters of samples (or conditions) are assigned to one context. Clustering samples or conditions in addition to variables can guide experimental design for validation of regulations [7].

2.1 Modeling Object Variables

We model all observed variables for objects $\{1, ..., N\}$ 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 objects. The covariance and mean capture two different aspects of the model regarding regulatory wiring and context-specific programs, 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

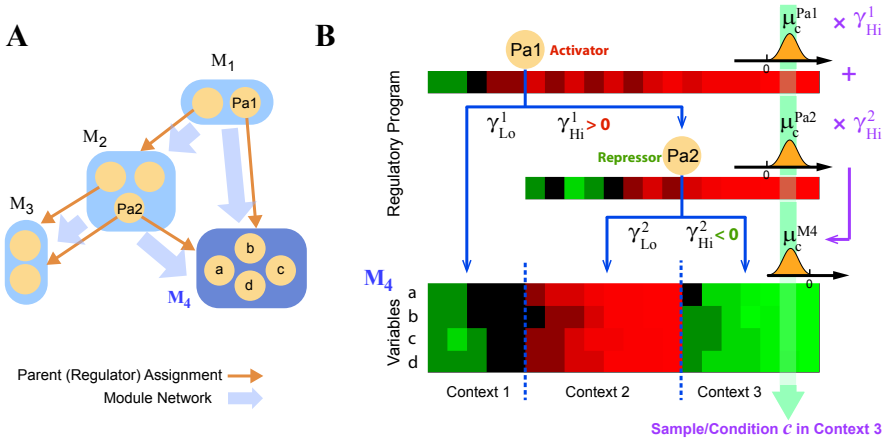

Figure 1: **(A)** Example module network **(B)** A combinatorial regulatory program is inferred for each module; example shown for M_4.

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 $\mathbf{\Sigma}$. To construct this structure, we relate object 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 $\mathbf{\Sigma} = (I-W)^{-T}(I-W)^{-1}$. Therefore, we impose the modular dependency structure over Σ through W, which is easier to interpret based on $\mathcal{A}, \mathcal{S}$ assignments.

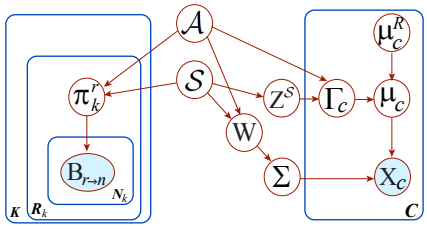

Figure 2: Graphical representation of model

We define variable means $\boldsymbol{\mu}_c$, based on parents as described below. First, based on the modular structure of genes, 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 objects in module k. In modules where there is more than one parent assigned, different states of parents, creating a context, can lead to different mechanisms of combinatorial regulation. 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 states are represented as a decision tree for each module k (figure 1.B). Thus, we represent a context-specific program as a unique dependency of variable means on parents, such that $\boldsymbol{\mu}_c^k$ for module k is a linear mixture of means for parents of that module: $\boldsymbol{\mu}_c^k = \sum_{r=1}^{R_k} \gamma_c^r \boldsymbol{\mu}_c^{Pa_k}$ where R_k is the number of parents Pa_k and γ_c^r are similar for all conditions c occurring in the same context. Thus, in general we can write $\boldsymbol{\mu}_c = \Gamma_c \boldsymbol{\mu}_c^R$, where $\boldsymbol{\mu}_c^R$ contains the means of regulators $1, ..., R$ 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 denoted the set of parent split-points for all modules. Then for each condition we have: $P(\mathbf{X_c}|\mathcal{A}, \mathcal{S}, \theta_c, \Sigma, Z^S) = \frac{1}{(2\pi)^{N/2}|\mathbf{\Sigma}^{1/2}|} exp(-\frac{1}{2}(\mathbf{X_c} - \boldsymbol{\mu}_c)^T \mathbf{\Sigma}^{-1}(\mathbf{X_c} - \boldsymbol{\mu}_c))$.

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 regulators in

different contexts (partitions of conditions), this leads to a bias or large signal variations in observed 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 Relational Data

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

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

The parameter π_k^r defines the probability of parent r interacting (and influencing) module M_k (figure 2). In the gene regulation 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-ChIP or ChIP-Seq technologies. Therefore, directed interactions from parents to all objects 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 objects we have:

$$P(\mathbf{B} | \mathcal{A}, S, \boldsymbol{\pi}) = \prod_{k=1}^{K} \prod_{r \in Pa_k} \prod_{n \in M_k} P(B_{r \to n} | \mathcal{A}, \mathcal{S}, \pi_k^r) \tag{2}$$

$$= \prod_{k=1}^{K} \prod_{r \in Pa_k} (\pi_k^r)^{s_{rk}} (1 - \pi_k^r)^{|M_k| - s_{rk}} \prod_{r' \notin Pa_k} (\pi_0)^{s_{r'k}} (1 - \pi_0)^{|M_k| - s_{r'k}} \tag{3}$$

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 relational data model and $|M_k|$ is the number of objects 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 effect 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 objects.

The likelihood of the joint model $\mathcal{M} = \{\mathcal{A}, \mathcal{S}, \Theta, \Sigma, Z^S, \boldsymbol{\pi}\}$ given the integration of variable and interaction 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: Identifiability of Joint Model

Our method uses relational 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 joint learning proposed in this paper resolves the un-identifiability issue. First, we show that modeling object variable alone is identifiable only under very specific conditions. Then, we will restate some results from [15] on the identifiability of overlapping block models. Using this result we show the identifiability of the joint model under some reasonable conditions.

Lemma 1. Variable data Model: *For the model of observed 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 objects and* $\sum_k |Pa_k| < N$ *and the covariance matrix* Σ *is invertible, we can conclude:* $\Theta = \Theta'$*,* $W = W'$ *(Proof sketch in Appendix).*

The above lemma provides identifiability for the case where the structure $\{\mathcal{A}, \mathcal{S}\}$ is assumed to be known. However, in the case that 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 following, we will use a theorem for identifiability of overlapping block models from [15] which is an extension of the results in [3]. The results provide conditions for overlapping stochastic block models to be identifiable.

Theorem 1. Relational 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 sketch in Appendix).*

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

Theorem 2. Identifiability of the joint 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 objects 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 sketch in Appendix).*

This Theorem, states the theoretical effect of joint modeling on identifiability of modular structures, given that commonly the sum of number of parents are less than the number of objects (as in gene regulatory networks).

4 Reversible Jump MCMC for Parameter Estimation

We use a Gibbs sampler to obtain the joint 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 [11] 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 regulators assigned to the same module. This set of distint regulator means $\boldsymbol{\mu}_c^{\mathbf{R}}$ are sampled with a normal proposal $Q_\mu(\boldsymbol{\mu}_c^{\mathbf{R}^{(i+1)}}|\boldsymbol{\mu}_c^{\mathbf{R}^{(i)}}) \sim \mathcal{N}(\boldsymbol{\mu}_c^{\mathbf{R}^{(i)}}, I)$. The means of all variables $\boldsymbol{\mu}_c^{(i+1)}$ and $\Theta^{(i+1)}$ are then computed accordingly. 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 also through a normal proposal. Due the symmetric proposal distribution, the proposal is accepted with probability $P_{mh} = \min\{1, \frac{P(\mathcal{M}^{(i+1)}|X,B)}{P(\mathcal{M}^{(i)}|X,B)}\}$ where $\mathcal{M}^{(i)} = \{\mathcal{A}, \mathcal{S}, \Theta, \Sigma, Z^S\boldsymbol{\pi}\}^{(i)}$.

4.2 Learning Module Assignment $\mathcal{A}$

Learning the assignment of each object 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 [11], an extension of the Metropolis-Hastings algorithm that allows moves between models with different dimensionality. In each proposal, we consider three close move schemes on assignment function $\mathcal{A}$: increasing or decreasing the number of modules by one, or not changing the total number. For increasing the number of modules, a random object is moved to a new module and for decreasing the number, two modules are merged. In the third case, an object is randomly moved from one module to another module. We design transformation of parameters using Green's method to extend model dimensions (Algorithm 1) The acceptance ratio for the split move is $P_{split} = \min\{1, \frac{P(\mathcal{M}^{(i+1)}|X,B)}{P(\mathcal{M}^{(i)}|X,B)} \times \frac{\frac{1}{K+1}}{\frac{1}{K}} \times \frac{p_{+1}}{p_{-1}} \times \frac{1}{p(\mathbf{u})p(\mathbf{u}')} \times \mathcal{J}_{(i)\to(i+1)}\}$ where $\mathcal{J}_{(i)\to(i+1)}$ is the Jacobian of the transformation from the previous state to the proposed state, and the acceptance ratio for the merge move is $P_{merge} = \min\{1, \frac{P(\mathcal{M}^{(i+1)}|X,B)}{P(\mathcal{M}^{(i)}|X,B)} \times \frac{\frac{1}{K-1}}{\frac{1}{K}} \times \frac{p_{-1}}{p_{+1}} \times \mathcal{J}_{(i)\to(i+1)}\}$.

4.3 Learning Dependency Structure $\mathcal{S}$

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 2). In the case of addition of a parent to a module, we propose mixture coefficients γ and interaction parameters π for the added regulator, based on its learned values in another module, where it has already been assigned as a parent, with an

Algorithm 1 RJMCMC to update $\mathcal{A}$

Find K: number of distinct modules in $\mathcal{A}^{(i)}$
Propose move ν from $\{-1, 0, +1\}$ with probabilities p_{-1}, p_0, p_{+1}, respectively.
switch (ν)
case +1**:**
  Select random object $n \in M_k$ uniformly
  Assign n to new module M_{K+1}
  Assign parents $Pa_{K+1} = Pa_k$
  Draw vectors $\mathbf{u}, \mathbf{u}' \sim \mathcal{N}(0, 1)$
  Propose parameters:
  $\boldsymbol{\pi}_{k1}^{Pa_{K+1}} = \boldsymbol{\pi}_k^{Pa_k} - \mathbf{u}$, $\boldsymbol{\pi}_{k2}^{Pa_k} = \boldsymbol{\pi}_k^{Pa_k} + \mathbf{u}$
  $\boldsymbol{\gamma}_{k1}^{Pa_{K+1}} = \boldsymbol{\gamma}_k^{Pa_k} - \mathbf{u}'$, $\boldsymbol{\gamma}_{k2}^{Pa_k} = \boldsymbol{\gamma}_k^{Pa_k} + \mathbf{u}'$
  Compute $\{\Theta, \Sigma, \boldsymbol{\pi}\}$
  Accept $\mathcal{A}^{(i+1)}$ with P_{split}
case -1**:**
  Select two random modules M_{k_1} and M_{k_2}
  Merge into one module $M_k 1$
  Assign parents $Pa_{k1} = Pa_{k_1} \cup Pa_{k_2}$
  for $\forall r \in Pa_{k_1} \cap Pa_{k_2}$ **do**
    Propose $\boldsymbol{\pi}_{k1}^r = (\boldsymbol{\pi}_{k1}^r + \boldsymbol{\pi}_{k2}^r)/2$
    and $\boldsymbol{\gamma}_{k1}^r = (\boldsymbol{\gamma}_{k1}^r + \boldsymbol{\gamma}_{k2}^r)/2$
  end for
  Compute $\{\Theta, \Sigma, \boldsymbol{\pi}\}$
  Accept $\mathcal{A}^{(i+1)}$ with P_{merge}
case 0**:**
  Select two random modules M_{k_1}, M_{k_2}
  Move a random object n from M_{k_1} to M_{k_2}
  Compute $\{\Theta, \Sigma, \boldsymbol{\pi}\}$
  Accept $\mathcal{A}^{(i+1)}(n) = k_2$ with P_{mh}
end switch

Algorithm 2 RJMCMC to update $\mathcal{S}$

Set p_S
for module $k = 1$ **to** K **do**
  Propose ν from $\{+1, -1\}$ with p_S
  switch (ν)
  case $+1$**:**
    Add a random parent $r \in 1, ..., R$ to Pa_k
    Draw $u, \mathbf{u}' \sim Unif(0, 1)$
    if r is also a parent of another module $Pa_{k'}$ **then**
      Propose $\pi_k^r = \pi_{k'}^r + u$, $\gamma_c^{r_k} = \gamma_c^{r_{k'}} + \mathbf{u}'(c)$ for all $c \in \{1, ..., C\}$
    else
      Propose $\pi_k^r = u$,$\gamma_c^{r_k} = \mathbf{u}'(c)$ for all c
    end if
    Compute $\{\Theta, \Sigma, \boldsymbol{\pi}\}$
    Accept $\mathcal{S}^{(i+1)}$ with P_{add}
  case -1**:**
    Remove a random parent r from Pa_k
    Compute $\{\Theta, \Sigma, \boldsymbol{\pi}\}$
    Accept $\mathcal{S}^{(i+1)}$ with P_{rem}
  end switch
end for

additional noise term. The acceptance ratio for the add proposal is $P_{add} = \min\{1, \frac{P(\mathcal{M}^{(i+1)}|X,B)}{P(\mathcal{M}^{(i)}|X,B)} \times \frac{\frac{1}{R_k+1}}{\frac{1}{R-R_k}} \times \frac{p_S}{1-p_S} \times \frac{1}{p(u)p(\mathbf{u}')} \times \mathcal{J}_{(i)\to(i+1)}\}$ where R_k is the number of parents for module k in the $i-$th state, and the acceptance ratio for the remove proposal is $P_{rem} = \min\{1, \frac{P(\mathcal{M}^{(i+1)}|X,B)}{P(\mathcal{M}^{(i)}|X,B)} \times \frac{\frac{1}{R-R_k+1}}{\frac{1}{R_k}} \times \frac{1-p_S}{p_S} \times \mathcal{J}_{(i)\to(i+1)}\}$.

5 Results

5.1 Synthetic Data

We first tested our method on synthetic variable and interaction data generated from the proposed model. A dataset was generated for $N = 200$ objects in $K = 4$ modules with $C = 50$ conditions for each object variable. Parents were assigned from a total of $R = 10$ number of candidate regulators. 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 expression 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 regulatory 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 integrative model (figure 3). As comparison, for each generated dataset, we also tested the sub-model for variable

data (excluding the model for interaction data) to infer regulatory links (figure 3). 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 integrative model compared to the expression 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.

5.2 Mycobacterium tuberculosis Gene Regulatory Network

We applied our method to identify regulatory modules for Mycobacterium tuberculosis (MTB). MTB is the causative agent of the tuberulosis disease in humans and the mechanisms underlying its ability to persist inside the host are only partially known [8]. We used interaction data identified from 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. [9]. We tested our method on 3072 MTB genes which had binding from at least one of the factors and performed 100,000 number of iterations on the combination of the two datasets. Out of the total genes, 815 could be assigned as a member of a module with high confidence (posterior probability of assignment> 0.9).

Figure 4(left) shows one of the identified modules, with an interesting regulatory program involving two MTB hypoxic adaptation regulators: Rv3133c (DosR) and Rv0081. Adaptation of MTB to hypoxia, i.e. oxygen deprivation is known to be an important factor for its persistense [8, 9]. DosR is well known to activate the initial response of MTB in hypoxic coniditions [18], and Rv0081 has recently been identified as a major hub in the MTB ChIP-Seq network [9]. The inferred regulatory program for this module predicts induction of the module in conditions where both of these factors are expressed (context (c) in figure 4). Rv0081 itself is also regulated by DosR, which creates a feed-forward loop structure driving this module (see figure 4). The genes assigned to this module include regulators known to be induced in later time points (after 24 hours) in hypoxia and this prediction illustrates the significance of Rv0081 in mediating the enduring response. Figure 4 (right) shows the global regulatory structures between the largest identified modules highlighting major MTB regulators. For modules with a single regulator, activation or repression signs were inferred based on estimated coefficients γ^r. We also found functional enrichment of largest modules using Gene Ontology (GO) terms and COG category annotations from the TBDB database [10, 20] in table 1.

As comparison, we applied the module networks method on the above expression data. We set the maxmimum number of modules to 10 and candidate pool of regulators to the 50 ChIPped regulators only. The method identified 2401 TF-gene interactions, out of which only 215 (8.9%) had ChIP evidence for binding to upstream or genic regions of genes and only 18.53% of genes had binding from at least one of the regulators assigned to their module. For a fair comparison of models without

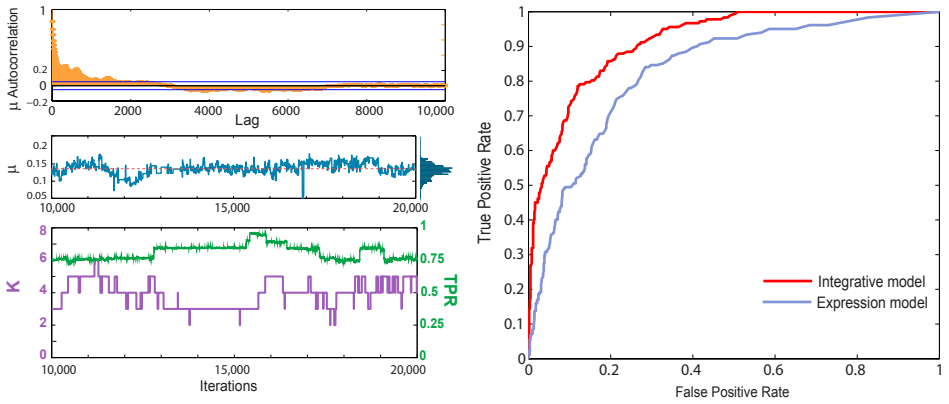

Figure 3: Autocorrelation for an example variable mean (top); gibbs samples and posterior after burn-in period (actual value shown with red line); number of modules (purple) and true positive rate of recovered dependencies (green), ROC curve for integrative model and expression model (right)

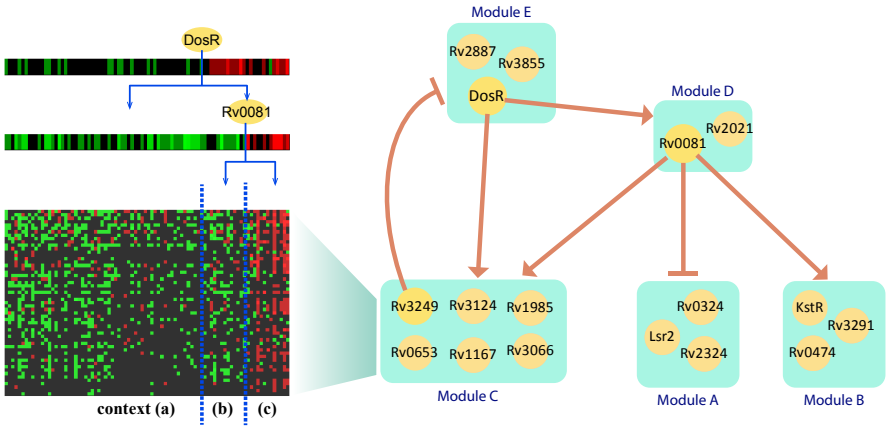

Figure 4: (Left) Result module showing combinatorial regulation between major TB hypoxic response regulators DosR and Rv0081. (Right) Global interactions between largest modules highlighting containing regulators

ID	Module Regulators	Enriched COG Catergories ($p < 0.05$)	Enriched GO terms ($p < 0.05$)
A	Rv0081 (Repressor)	Energy production and conversion; Translation, ribosomal structure and biogenesis; Amino acid transport and metabolism; Replication, recombination and repair	NADH dehydrogenase activity; extracellular region; growth; nitrate reductase activity; plasma membrane; cell wall
B	Rv0081 (Activator)	Cell wall/membrane/envelope biogenesis; Secondary metabolites biosynthesis, Carbohydrate transport and metabolism;	sulfolipid biosynthetic process; growth
C	DosR, Rv0081	Inorganic ion transport and metabolism; Transcription	growth
D	DosR	Posttranslational modification, protein turnover, chaperones; Signal transduction mechanisms	cellular response to nitrosative stress
E	Rv3249	Carbohydrate transport and metabolism; Coenzyme transport and metabolism; Translation, ribosomal structure and biogenesis; Amino acid transport and metabolism	plasma membrane; extracellular region
F	KstR	Secondary metabolites biosynthesis, transport and catabolism; Lipid transport and metabolism	cytosol

Table 1: Enrichment of functional annotations for largest identified modules in MTB network

the effect of interaction data, we also applied our model for expression data only. As a result, 4264 interactions were identified, out of which 739 (17.33%) had binding evidence, and 32.76% of genes had binding from at least one of the regulators assigned to their module. Thus, module networks and in general models based on co-expression data infer mostly indirect (or correlated) regulators and these results clarify the significance of using interaction data and tools for integrating interaction data with expression data (or other complementary data types), for inference of direct regulatory relationships.

6 Conclusion

We proposed a model for learning dependency structures between modules of objects, by joint learning from relational data and object variable data. This integration improves accuracy and avoids over-fitting. We presented a reversible-jump inference procedure for learning model posterior which can be interpreted based on context. Our results showed high performance on both synthetic and genetic data. The framework allows integration of other data types, including expert knowledge by imposing prior distributions.

References

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

[2] E.M. Airoldi, D.M. Blei, S.E. Fienberg, E.P. Xing, and T. Jaakkola. Mixed membership stochastic block models for relational data with application to protein-protein interactions. In *Proceedings of the Int. Biometrics Society Annual Meeting*, 2006.

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

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

[5] A. Bernard, A.J. Hartemink, et al. Informative structure priors: joint learning of dynamic regulatory networks from multiple types of data. In *Pac Symp Biocomput*, volume 10, pages 459–470, 2005.

[6] N.E. Buchler, U. Gerland, and T. Hwa. On schemes of combinatorial transcription logic. *Proceedings of the National Academy of Sciences*, 100(9):5136–5141, 2003.

[7] R. De Smet and K. Marchal. Advantages and limitations of current network inference methods. *Nature Reviews Microbiology*, 8(10):717–729, 2010.

[8] JoAnne L Flynn and John Chan. Tuberculosis: latency and reactivation. *Infection and immunity*, 69(7):4195–4201, 2001.

[9] James E Galagan, Kyle Minch, Matthew Peterson, Anna Lyubetskaya, Elham Azizi, Linsday Sweet, Antonio Gomes, Tige Rustad, Gregory Dolganov, Irina Glotova, et al. The mycobacterium tuberculosis regulatory network and hypoxia. *Nature*, 499(7457):178–183, 2013.

[10] James E Galagan, Peter Sisk, Christian Stolte, Brian Weiner, Michael Koehrsen, Farrell Wymore, TBK Reddy, Jeremy D Zucker, Reinhard Engels, Marcel Gellesch, et al. Tb database 2010: overview and update. *Tuberculosis*, 90(4):225–235, 2010.

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

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

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

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

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

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

[17] T. Michoel, S. Maere, E. Bonnet, Y. Joshi, A.and Saeys, T. Van den Bulcke, K. Van Leemput, P. Van Remortel, M. Kuiper, K. Marchal, et al. Validating module network learning algorithms using simulated data. *BMC Bioinformatics*, 8(Suppl 2):S5, 2007.

[18] Heui-Dong Park, Kristi M Guinn, Maria I Harrell, Reiling Liao, Martin I Voskuil, Martin Tompa, Gary K Schoolnik, and David R Sherman. Rv3133c/dosr is a transcription factor that mediates the hypoxic response of mycobacterium tuberculosis. *Molecular microbiology*, 48(3):833–843, 2003.

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

[20] TBK Reddy, Robert Riley, Farrell Wymore, Phillip Montgomery, Dave DeCaprio, Reinhard Engels, Marcel Gellesch, Jeremy Hubble, Dennis Jen, Heng Jin, et al. Tb database: an integrated platform for tuberculosis research. *Nucleic acids research*, 37(suppl 1):D499–D508, 2009.

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

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

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

[24] A.V. Werhli and D. Husmeier. 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.