Latent Variable Model for Learning in Pairwise Markov Networks

Saeed Amizadeh[1] and **Milos Hauskrecht**[1,2]
[1]Intelligent Systems Program, [2]Computer Science Department,
University of Pittsburgh, 210 S. Bouquet St.
Pittsburgh, PA 15260, USA

Abstract

Pairwise Markov Networks (PMN) are an important class of Markov networks which, due to their simplicity, are widely used in many applications such as image analysis, bioinformatics, sensor networks, etc. However, learning of Markov networks from data is a challenging task; there are many possible structures one must consider and each of these structures comes with its own parameters making it easy to overfit the model with limited data. To deal with the problem, recent learning methods build upon the L1 regularization to express the bias towards sparse network structures. In this paper, we propose a new and more flexible framework that let us bias the structure, that can, for example, encode the preference to networks with certain local substructures which as a whole exhibit some special global structure. We experiment with and show the benefit of our framework on two types of problems: learning of modular networks and learning of traffic networks models.

Introduction

Pairwise Markov Networks (PMN) define an important class of probabilistic graphical models (Koller and Friedman, 2009) with applications in areas as diverse as image analysis, genetics, natural language processing or transportation network analysis. PMNs represent the structure of the joint distribution in terms of direct interactions (dependencies) among individual variables and *potential* functions for each interacting pair. In general, learning of PMNs from data is a hard task. The parameters of the network with a fixed structure are found using iterative methods such as iterative proportional scaling (Jirousek and Přeučil, 1995) and its variants. The learning of the structure or the learning of the complete Markov network (both the parameters and the structure) is even harder. However, learning PMNs may be particularly important for knowledge discovery; for example, the automatic learning of the interaction structure among genes from observational data may provide new insights into the cell regulatory mechanisms.

The structure learning problem for PMNs has been the topic of active research by AI and UAI communities in recent years. The methods typically applied to this problem, rely on the L1 (lasso) regularization (Lee, Ganapathi, and Koller (2007), Wainwright, Ravikumar, and Lafferty (2007), Friedman, Hastie, and Tibshirani (2008), Ambroise, Chiquet, and Matias (2009), Banerjee, Ghaoui, and d'Aspremont (2008), Meinshausen and Bühlmann (2006), Marlin and Murphy (2009)) which optimizes the loglikelihood of data penalized with the L1 regularization term. The L1 regularization term biases the learning towards sparser structures and in principle defines a prior distribution over possible PMN structures. However, this approach lacks the flexibility of defining other types of structural biases. To address this limitation, we propose a new PMN learning method that lets us incorporate a larger class of structural priors over possible models into learning process. The priors, induced by the L1 regularization penalty, then become a special case. In our model, the prior distribution over the structures is defined in terms of a joint probability distribution over a set of latent variables, each representing the presence or the absence of one edge in the underlying PMN model. This higher level model and its distribution is defined by another (structure-generating) Markov network and its corresponding *energy functions*. The structure and the parameter learning of the underlying PMN model are then solved by optimizing the loglikelihood of the data using the variational expectation-maximization (EM) approach (Jordan et al., 1998) on the extended model.

Our proposed framework for learning PMNs is very general and may be customized to any type of PMN (for example, a binary PMN (Höfling and Tibshirani, 2009) or a Gaussian graphical model (Friedman, Hastie, and Tibshirani, 2008)), any type of energy functions defining the structural prior, or additional regularization penalties (L1, L2). In this paper, due to the space limitation, we experiment with pairwise Gaussian Graphical Models (GGMs), L1 regularization terms, and two types of energy functions defining structural priors. We show that these energy functions are able to recover the structure of the underlying GGM better than other structure-learning methods.

The structure of the paper is as follows. First we introduce a latent variable model with a structure-generating layer defining the structural prior. After that we describe a variational EM method we use to solve the latent variable model. Then we briefly describe the options for refining our general framework. Finally we test our method on two learning problems: (1) learning of modular PMNs, and (2)

learning of vehicular traffic networks.

Latent Variable Model for Learning PMNs

A pairwise Markov network (PMN) $\boldsymbol{\mathcal{M}}$ is a probabilistic graphical model over a set of random variables $\mathbf{X} = \{X_1, X_2, \ldots, X_p\}$ that is defined by an undirected graph $\boldsymbol{\mathcal{G}} =< \mathbf{X}, \mathbf{E} >$ and two sets of *potentials*: node potentials $\boldsymbol{\mathcal{V}} = \{v_i \mid i \in 1..p\}$, and edge potentials $\boldsymbol{\mathcal{F}} = \{f_{ij} \mid \{i,j\} \in \mathbf{E}\}$. Here, $\mathbf{E}$ is a subset of all possible undirected edges $\mathbf{L}$ defined over the set of nodes (variables) in $\mathbf{X}$. Each edge potential f_{ij} is a parametric positive function over X_i and X_j parameterized by the vector $\boldsymbol{\theta_{ij}}$, i.e. $f_{ij} \triangleq f(X_i, X_j; \boldsymbol{\theta_{ij}}) > 0$. The same holds for node potentials; that is, $v_i \triangleq v(X_i; \boldsymbol{\theta_i}) > 0$. The distribution over $\mathbf{X}$ is subsequently defined as (Koller and Friedman, 2009):

$$\Pr(\mathbf{X} \mid \boldsymbol{\Theta}) = \frac{1}{\alpha(\boldsymbol{\Theta})} \prod_{\{i,j\} \in \mathbf{E}} f(X_i, X_j; \boldsymbol{\theta_{ij}}) \prod_{i=1}^{p} v(X_i; \boldsymbol{\theta_i}), \tag{1}$$

where, $\alpha(\boldsymbol{\Theta})$ is the normalization factor (aka the *partition function*) and $\boldsymbol{\Theta} = \{\boldsymbol{\theta_{ij}} \mid \{i,j\} \in \mathbf{E}\} \cup \{\boldsymbol{\theta_i} \mid i \in 1..p\}$.

Having formalized PMNs, the task of learning a PMN from data is defined as follows: given a set of observations $\boldsymbol{\mathcal{D}}$ over variables $\mathbf{X}$ drawn independently from some underlying PMN $\boldsymbol{\mathcal{M}}$, find an edge set $\mathbf{E}^*$ (the structure) and its corresponding parameter set $\boldsymbol{\Theta}^*$ so that the PMN $\boldsymbol{\mathcal{M}}^*$, induced by $\mathbf{E}^*$ and $\boldsymbol{\Theta}^*$, is as *close* to $\boldsymbol{\mathcal{M}}$ as possible. We approach this problem by formalizing the problem probabilistically with the help of a group of latent variables representing the edges between the variables in the model and the prior distribution over different structures.

Dummy potentials and parameter densities: We assume there exists a specific parameter vector $\boldsymbol{\theta}_0$ such that $f(X_i, X_j; \boldsymbol{\theta}_0) = 1$ for all possible values of X_i and X_j. Therefore, we can extend $\boldsymbol{\mathcal{F}}$ to contain edge potentials over *all* possible pairs such that the edge potentials for missing edges are parameterized by $\boldsymbol{\theta}_0$. The potentials for missing edges are called *dummy potentials*. Furthermore, we assume the edge parameter vectors for existing edges are drawn from some density P_e. For missing edges, the parameter vector is ideally equal to $\boldsymbol{\theta}_0$. However, due to the finite sample size and presence of noise, we model the parameter vectors for missing edges to be drawn from some sharp density P_m centered on $\boldsymbol{\theta}_0$.

Auxiliary latent variables: For each candidate edge e_{ij} (for a pair of variables X_i and X_j), we define an (auxiliary) latent binary variable Z_{ij} to explicitly model the presence or absence of that edge in the PMN model. The value of Z_{ij} indicates which distribution the parameter vector $\boldsymbol{\theta_{ij}}$ is drawn from. $Z_{ij} = 1$ means that e_{ij} is present and $\boldsymbol{\theta_{ij}}$ is drawn from P_e while $Z_{ij} = 0$ implies that e_{ij} is absent and $\boldsymbol{\theta_{ij}}$ is drawn from P_m. Subsequently, we will have $P_e = \Pr(\boldsymbol{\theta_{ij}} | Z_{ij} = 1)$ and $P_m = \Pr(\boldsymbol{\theta_{ij}} | Z_{ij} = 0)$. We call the binary matrix $\mathbf{Z} = [Z_{ij}]_{p \times p}$ *the structural matrix* since it encodes the structure of the model.

By extending $\boldsymbol{\Theta}$ to contain the parameter vectors for missing edges as well (which are ideally equal to $\boldsymbol{\theta}_0$), knowing $\boldsymbol{\Theta}$ will completely specify the model to generate data. In other words, the extended set $\boldsymbol{\Theta}$ implicitly encodes the structure as well. As a result, given $\boldsymbol{\Theta}$, the dataset $\boldsymbol{\mathcal{D}}$ will become independent of the structural matrix $\mathbf{Z}$. Why do we need $\mathbf{Z}$ then? As we will see next, by imposing some dependency structures on binary variables Z_{ij}s, we can introduce some biases toward some desired structures. Note that, an alternative model is to make data directly dependent on $\mathbf{Z}$ without introducing dummy potentials. The problem with this model is that the partition function is dependent on $\mathbf{Z}$ which makes the computations intractable.

Structural priors: The prior probability of different PMN structures can be defined in terms of the joint probability distribution over the latent variables $\mathbf{Z}$. We define the structural prior over $\mathbf{Z}$ as:

$$\Pr(\mathbf{Z}) = \frac{1}{\alpha_\phi} \exp\big(-\phi(\mathbf{Z})\big) \tag{2}$$

Here, α_ϕ is the normalization constant and $\phi(\cdot)$ is a non-negative *energy function* which maps a candidate structure $\mathbf{Z}$ to a real energy value. $\phi(\cdot)$ is designed in a way that more preferable (stable) structures get lower energy values. The different choices of the energy function will be discussed later in this paper.

Having defined all the elements of our generative model, now we derive the joint distribution $\Pr(\boldsymbol{\mathcal{D}}, \boldsymbol{\Theta}, \mathbf{Z})$ in our model. Assuming an I.I.D. sample $\boldsymbol{\mathcal{D}} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}$, the log-joint distribution over data $\boldsymbol{\mathcal{D}}$, the latent variables $\mathbf{Z}$, and the parameters $\boldsymbol{\Theta}$ can be expressed as:

$$\begin{aligned} \log \Pr(\boldsymbol{\mathcal{D}}, \boldsymbol{\Theta}, \mathbf{Z}) &= \sum_{i=1}^{n} \log \Pr(\mathbf{x}^{(i)} \mid \boldsymbol{\Theta}) - \phi(\mathbf{Z}) \\ &+ \sum_{i \neq j} \log \Pr(\boldsymbol{\theta_{ij}} \mid Z_{ij}) + \sum_{i=1}^{p} \log \Pr(\boldsymbol{\theta_i}) - \log \alpha_\phi \end{aligned} \tag{3}$$

Variational Learning Algorithm

Given the expanded model with the latent variables $\mathbf{Z}$ that represent the edges, our goal is to estimate $\boldsymbol{\Theta}$ that maximizes the posterior distribution $\Pr(\boldsymbol{\Theta} \mid \boldsymbol{\mathcal{D}}) \propto \Pr(\boldsymbol{\Theta}, \boldsymbol{\mathcal{D}})$. However, since we have a latent component in our model (i.e. $\mathbf{Z}$), we cannot directly maximize $\Pr(\boldsymbol{\Theta}, \boldsymbol{\mathcal{D}})$. Instead, we can, in principle, carry this optimization using the Expectation-Maximization (EM) algorithm that solves the problem iteratively by taking the expectation of Equation (3) over $\mathbf{Z}$ with respect to $\Pr(\mathbf{Z} \mid \boldsymbol{\Theta}, \boldsymbol{\mathcal{D}}) \equiv \Pr(\mathbf{Z} \mid \boldsymbol{\Theta})$ (due to $\mathbf{Z} \perp \boldsymbol{\mathcal{D}} \mid \boldsymbol{\Theta}$) in the E-step and maximizing the result over $\boldsymbol{\Theta}$ in the M-step. However, due to the complex form of $\Pr(\mathbf{Z} \mid \boldsymbol{\Theta})$ in our model, taking the expectation is computationally very costly and, in general, requires us to sum over exponentially many assignments of values to $\mathbf{Z}$. To alleviate the problem, we adopt the variational EM method (Jordan et al., 1998) that approximates $\Pr(\mathbf{Z} \mid \boldsymbol{\Theta})$ with a different, but computationally more convenient distribution. Using this technique, not only we estimate $\boldsymbol{\Theta}$, but also, as a byproduct, we get a factorized approximation to $\Pr(\mathbf{Z} \mid \boldsymbol{\Theta})$ which can be used to infer the structure of the underlying model very efficiently.

Mean Field Approximation of $\Pr(\mathbf{Z} \mid \mathbf{\Theta})$

Our goal is to find a distribution $Q(\mathbf{Z})$ that approximates $\Pr(\mathbf{Z} \mid \mathbf{\Theta})$ well and at the same time permits efficient inferences in the E-step. To find this distribution we resort to the mean field approximation (Jordan et al., 1998):

$$Q(\mathbf{Z}) = \prod_{i \neq j} q_{ij}(Z_{ij}) = \prod_{i \neq j} \delta_{ij}^{Z_{ij}} (1 - \delta_{ij})^{1 - Z_{ij}} \quad (4)$$

where, δ_{ij}s are the parameters of our approximation. Note that parameters δ_{ij} are free (variational) parameters that can be optimized to better approximate $\Pr(\mathbf{Z} \mid \mathbf{\Theta})$. Also, note that $Q(\mathbf{Z})$ nicely decomposes along variables Z_{ij}s.

Variational Lower Bound

To optimize both $\mathbf{\Theta}$ and $Q(\mathbf{Z})$, we maximize: :

$$\mathcal{J}(\boldsymbol{\mathcal{D}}, \mathbf{\Theta}, Q(\mathbf{Z})) = \mathbb{E}_Q\{\log \Pr(\boldsymbol{\mathcal{D}}, \mathbf{\Theta}, \mathbf{Z})\} + \mathcal{H}(Q(\mathbf{Z})) \quad (5)$$

where $\mathcal{H}(\cdot)$ denotes the entropy function. More precisely, we try to maximize the expected $\log \Pr(\boldsymbol{\mathcal{D}}, \mathbf{\Theta}, \mathbf{Z})$ with respect to a new distribution $Q(\mathbf{Z})$ instead of $\Pr(\mathbf{Z} \mid \mathbf{\Theta})$ whose entropy is high. It can be shown that:

$$\mathcal{J} = \log \Pr(\boldsymbol{\mathcal{D}}, \mathbf{\Theta}) - \mathrm{D}_{KL}(Q(\mathbf{Z}) \| \Pr(\mathbf{Z} \mid \mathbf{\Theta})) \quad (6)$$

which is the lower bound of $\log \Pr(\boldsymbol{\mathcal{D}}, \mathbf{\Theta})$. Therefore, maximizing $\mathcal{J}$ with respect to $\mathbf{\Theta}$ and $Q(\mathbf{Z})$ is equivalent to maximization of the lower bound of $\log \Pr(\boldsymbol{\mathcal{D}}, \mathbf{\Theta})$. The difference between $\mathcal{J}$ and $\log \Pr(\boldsymbol{\mathcal{D}}, \mathbf{\Theta})$ is exactly the KL divergence of $Q(\mathbf{Z})$ and $Pr(\mathbf{Z} \mid \mathbf{\Theta})$. In our model, $\mathcal{J}$ becomes:

$$\begin{aligned} \mathcal{J} = & \sum_{i=1}^{n} \log \Pr(\mathbf{x}^{(i)} \mid \mathbf{\Theta}) - \mathbb{E}_Q\big\{\phi(\mathbf{Z})\big\} - \log \alpha_\phi \\ & + \sum_{i \neq j} \mathbb{E}_Q\big\{ \log \Pr(\boldsymbol{\theta_{ij}} \mid Z_{ij})\big\} + \sum_{i=1}^{p} \log \Pr(\boldsymbol{\theta_i}) \\ & - \sum_{i \neq j} \big(\delta_{ij} \log \delta_{ij} + (1 - \delta_{ij}) \log(1 - \delta_{ij})\big) \end{aligned} \quad (7)$$

Note that $\mathcal{J}$ includes the original parameters $\mathbf{\Theta}$ and the variational parameters δ_{ij} $\forall i, j$. Both parameter sets can be maximized using the Variational EM algorithm. In the E-step, given the current estimate of parameter set $\hat{\mathbf{\Theta}}$, we find $Q(\mathbf{Z})$ that maximizes $\mathcal{J}$. The $\mathcal{J}$ is maximized whenever

$$\forall i \neq j : \delta_{ij} = \frac{\Omega(\hat{\boldsymbol{\theta}}_{\boldsymbol{ij}}, z = 1)}{\Omega(\hat{\boldsymbol{\theta}}_{\boldsymbol{ij}}, z = 1) + \Omega(\hat{\boldsymbol{\theta}}_{\boldsymbol{ij}}, z = 0)} \quad (8)$$

such that

$$\begin{aligned} \Omega(\hat{\boldsymbol{\theta}}_{\boldsymbol{ij}}, z) = & \Pr(\hat{\boldsymbol{\theta}}_{\boldsymbol{ij}} \mid Z_{ij} = z) \\ & \times \exp(-\mathbb{E}_Q\big\{\phi(\mathbf{Z}) \mid Z_{ij} = z\big\}) \end{aligned} \quad (9)$$

(8) defines the mean-field fixed-point equations for δ_{ij}s and can be solved iteratively. This corresponds to the E-step of the Variational EM. In the M-step, we try to find $\mathbf{\Theta}$ that maximizes $\mathcal{J}$ given the current estimate of $\hat{Q}(\mathbf{Z})$ (i.e. $\hat{\delta}_{ij}$s).

Refinement of The Framework

So far, we have developed a general framework for learning PMN from a finite sample. In order to refine our framework for specific applications, we need to specify three components in our model: (I) the energy function $\phi(\cdot)$ which expresses our structural bias in a specific problem, (II) $\Pr(\boldsymbol{\theta_{ij}} \mid Z_{ij})$ which defines how the edge parameter vectors are distributed in different edge states (as we will see in this section, the choice of $\Pr(\boldsymbol{\theta_{ij}} \mid Z_{ij})$, in fact, determines the type of regularization that shows up in the M-step, e.g. L1, L2), and (III) $\Pr(\mathbf{X} \mid \mathbf{\Theta})$ which specifies the type of PMN we are interested to learn. While the choices (I) and (II) affect the E-step, the choices (II) and (III) affect the M-step. In the next two subsections, we discuss some useful candidates for these three choices.

Structural Priors

The Equation (2) defines the prior distribution over the different PMNs structures in terms of an energy function $\phi(\cdot)$. Hence, $\phi(\cdot)$ biases the learning algorithm toward some desired structures by assigning them less energy values. However, from the computational perspective, we also need to assure the conditional expectation of $\phi(\cdot)$ in Equation (9) is efficiently computable.

To this end, we define the class of *additive* energy functions as a general subclass of energy functions. The idea is that the energy of a candidate structure can be decomposed into the energies of its $substructures$. A substructure s, in general, is defined as a subset of all possible edges (e.g. a collection of all edges that form a specific shape or share a node, etc.) Note that different substructures are not necessarily disjoint and can have some edges in common. As a result, we have $\phi(\mathbf{Z}) = \sum_{s \in \mathbf{S}} \psi(s)$ where $\mathbf{S}$ is the set of all *interesting* substructures and $\psi(\cdot)$ is the sub-energy function which computes the energy of a substructure. In this paper, we propose (and later experiment with) two additive energy functions which we describe next.

The simplest substructures are the single edges. We define the *pairwise* energy of the edge $\{i, j\}$ as:

$$\begin{aligned} \psi_p(Z_{ij}) &= -\big(Z_{ij} \log \beta_{ij} + (1 - Z_{ij}) \log(1 - \beta_{ij})\big) \\ \beta_{ij} &= \Pr(Z_{ij} = 1) \end{aligned} \quad (10)$$

Here, β_{ij} expresses the prior belief regarding the existence of edge $\{i, j\}$ in the model. Using this energy function, it is easy to see that (8) reduces to the standard Bayesian formula for computing the posterior probability of the edge's existence (i.e. δ_{ij}). As expected, the pairwise energy function (PEF) can only be used in those problems where a good estimate of the edge priors is available beforehand. For example, in this paper (in the Experiments section), we employ our framework to learn a PMN over a set of road traffic sensors each of which has a geographical position and measures the traffic volume at that position. The intuition here is that geographically distant sensors are less likely to be directly correlated with each other. Therefore, the pairwise geographical distances can be used to form the edge priors. We will describe this problem in more details in the next section.

The second energy function we introduce here is called *clustering* energy function (CEF). The substructures for CEF are all sets of three edges which form a triangle. The energy of a triangle is then computed as:

$$\psi_c(Z_{ij}, Z_{jk}, Z_{ki}) = \epsilon \cdot I(Z_{ij}, Z_{jk}, Z_{ki}), \epsilon > 0 \qquad (11)$$

where, $I(\cdot,\cdot,\cdot)$ is an indicator function which returns 1 iff two of its inputs are 1 and the other one is 0. In other words, a triangle is assigned the energy of ϵ iff it is an $incomplete$ triangle, with two edges present and one missing. Using CEF, the conditional expectation in (9) can be derived as:

$$\begin{aligned} \mathbb{E}_Q\big\{\phi(\mathbf{Z}) \mid Z_{ij}=1\big\} &= \sum_{k\neq i,j} (\delta_{ik}+\delta_{jk}-2\delta_{ik}\delta_{jk}) \\ \mathbb{E}_Q\big\{\phi(\mathbf{Z}) \mid Z_{ij}=0\big\} &= \sum_{k\neq i,j} \delta_{ik}\delta_{jk} \end{aligned} \qquad (12)$$

The equalities in (12) are both up to a same constant c which is canceled out in (8). It turns out that CEF prefers clustered structures - those structures that consist of a couple of clusters each of which has dense connections inside and sparse connections with other clusters. This is in fact due to the penalizing of the incomplete triangles in a candidate structure by assigning ϵ energies to them which makes the whole structure unstable. Choosing greater values for ϵ indeed enforces more obvious clustered structure. Therefore, CEF is a perfect choice for those problems where the underlying PMN is believed to be *modular* meaning that variables are organized in a clustered structure. The experiments with CEF on modular problems are presented in the next section.

Gaussian Graphical Models

Gaussian Graphical Models (GGM) are a special class of PMN such that $\Pr(\mathbf{X} \mid \mathbf{\Theta})$ is a multi-variate Normal distribution (Koller and Friedman, 2009). If $\mathbf{K} = [k_{ij}]_{p\times p}$ is the positive definite precision matrix of this distribution and its mean vector is zero, the edge and the node potentials will be $f(X_i, X_j; k_{ij}) = \exp(-X_i k_{ij} X_j)$ and $v(X_i; k_{ii}) = \exp(-\frac{1}{2}k_{ii}X_i^2)$, respectively. Therefore, in GGMs, the edge and the node parameter vectors are actually the scalar values which are the elements of the precision matrix (i.e. $\boldsymbol{\theta_{ij}} = k_{ij}$, $\boldsymbol{\theta_i} = k_{ii}$). In the ideal case, if $\{i,j\} \notin \mathbf{E}$, we will have $k_{ij} = 0$ (that is, $\boldsymbol{\theta}_0 = 0$).

Modeling $\Pr(\boldsymbol{\theta_{ij}} \mid Z_{ij})$

The next choice is to specify $\Pr(\boldsymbol{\theta_{ij}} \mid Z_{ij})$ which in the GGM case corresponds to $\Pr(k_{ij} \mid Z_{ij})$. One option here is to adopt Laplace distribution:

$$\Pr(k_{ij} \mid Z_{ij}=t) = \frac{1}{2\lambda_t} \exp\left\{ -\frac{|k_{ij}-\mu_t|}{\lambda_t} \right\} \qquad (13)$$

where $t \in \{0,1\}$ and the hyper-parameters (μ_t, λ_t) represent the mean and the variance of the Laplace distribution, respectively. These hyper-parameters can be fixed or estimated in the E-step by taking derivatives of $\mathcal{J}$. In both cases, the only constraints are $\mu_0 = 0$ and $\lambda_1 > \lambda_0$. $\mu_0 = 0$ ensures the noise distribution $\Pr(k_{ij} \mid Z_{ij} = 0)$ is centered on $\boldsymbol{\theta}_0$. The second constraint expresses that the partial correlation coefficient k_{ij} has more variation if X_i and X_j are, in fact, directly correlated with each other (i.e. $\{i,j\} \in \mathbf{E}$).

Having specified choices (II) and (III), we can derive the M-step optimization problem for the GGMs with Laplace parameter distribution as follows. Assuming the current estimate of $\hat{Q}_\Theta(\mathbf{Z})$ and $\boldsymbol{\mathcal{D}}$, maximization of $\mathcal{J}$ over $\mathbf{K}$ is:

$$\hat{\mathbf{K}} \leftarrow \arg\max_{\mathbf{K}\succ 0} \big\{ \frac{n}{2}(\log\det\mathbf{K} - tr(\hat{\mathbf{\Sigma}}\mathbf{K})) - \|\mathbf{\Gamma}\star\mathbf{K}\|_1 \big\} \qquad (14)$$

$$\mathbf{\Gamma} = [\gamma_{ij}]_{p\times p},\ \gamma_{ij} = \frac{\delta_{ij}}{\lambda_1} + \frac{1-\delta_{ij}}{\lambda_0} \qquad (15)$$

where, n is the sample size, $\hat{\mathbf{\Sigma}}$ is the empirical covariance matrix of $\boldsymbol{\mathcal{D}}$, $tr(\cdot)$ is the trace of matrix, $\|\mathbf{A}\|_1 = \sum|a_{ij}|$ is the L1-norm of matrix $\mathbf{A}$, and $\star$ is the element-by-element matrix multiplication.

Interestingly, (14) can be viewed as the generalized L1-regularized maximum likelihood (L1ML) optimization for learning sparse GGMs. In fact, viewing $\mathbf{\Gamma}$ as the *penalty* matrix, the L1-norm in (14) would be a proxy for sparsity. The only difference is that, in the standard L1ML, uniform penalty values are multiplied by the elements of $\mathbf{K}$ (Friedman, Hastie, and Tibshirani, 2008), while in our case, the penalties for different k_{ij}s are different based on δ_{ij} values estimated in the E-step. As a matter of fact, the standard L1ML for learning sparse GGMs can be seen as a special case of our general framework in which all the edges *a priori* have the same probability of existence which determines the degree of sparseness. Equation (14) is a concave optimization problem which is guaranteed to have only one maximum and can be solved using standard *Block Coordinate Descent* algorithm used for the standard L1ML (Banerjee, Ghaoui, and d'Aspremont, 2008). Note that choice of Normal distribution instead of Laplace distribution for $\Pr(\boldsymbol{\theta_{ij}} \mid Z_{ij})$, in the previous subsection, would result in the L2 regularization in the M-step.

Experimental Results

In this section, we use the GGM refinement of our framework with two energy functions introduced in this paper to learn two different PMNs: modular networks and traffic sensor networks (where each variable is assigned a geographical position). The first experiment is performed on the synthetic data while the second one is performed on the real data.

Modular Networks: Modular networks are networks in which nodes are organized in some clusters each of which has dense connections inside and sparse connections with other clusters. Here, we apply our approach to data samples generated from a synthetic modular GGMs (with 200 variables organized in roughly 4 ring-shaped clusters). The objective is to uncover the original structure of the GGM (Figure 1(a)). In particular, we use CEF as the energy function for our framework which biases the structure learning process toward the modular structures. For this experiment, we generate training samples of different sizes, varying from 250 to 4250. The plots in Figure 1(b,c) represent the averages over 50 runs of the experiment with 2-standard deviation error bars. We compare our approach with three other methods developed for

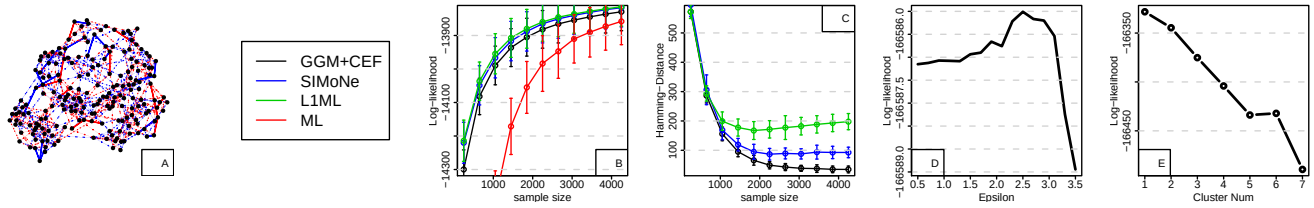

Figure 1: (a) A modular PMN (b) the log-likelihood of test data vs. sample size (c) the Hamming distances with the true model (d) penalized log-likelihood vs. ϵ (e) penalized log-likelihood vs. C

learning GGMs including non-regularized maximum likelihood (ML), standard L1-regularized maximum likelihood (L1-ML)(Banerjee, Ghaoui, and d'Aspremont, 2008), and SIMoNe (Ambroise, Chiquet, and Matias, 2009). SIMoNe along with the method in Marlin and Murphy (2009) are specialized algorithms to learn modular GGMs by first clustering the variables of the GGM and then applying the group L1 on the clustered structure.

We evaluate all the methods based on their generalization power as well as their structural accuracy. The score measuring the generalization power is the log-likelihood of a fixed test sample of size 100. Figures 1(b) depicts the average log-likelihood of the test data against the training sample size. These plots show that all the regularization methods including ours achieve pretty much the same performance in terms of the generalization power which is significantly better than that of the non-regularized ML. It should be noted that the mean generalization power for L1ML and SIMoNe is slightly better than that of our approach. However, the difference is insignificant.

To measure the structural accuracy, we compute the Hamming distance between the resulted structures and the true model (the number of differences between two structures). Figures 1(c) plots the Hamming distance against the sample size for all the regularization methods. The Hamming distances for the non-regularized ML were so large that we did not even include the corresponding curve in the plot. According to these results, as the sample size grows, the GGM realization of our framework with CEF achieves the most accurate structure (the lowest Hamming distance) with significant margins to L1ML and SIMoNe. However, for small sample sizes all of these methods except the non-regularized ML perform similarly in terms of the structural accuracy.

Note that SIMoNe, as a specialized method for learning modular networks, needs to know the true number of clusters C in advance. For CEF, on the other hand, the value of ϵ energy assigned to incomplete triangles should be given initially. In principle, we can find the values of these parameters using cross-validation: Figure 1(d,e) shows the penalized log-likelihood of test data for different values of ϵ and C. However, ϵ is conceptually very different from C. Regardless of the true C, by choosing large ϵ values, we bias our algorithm toward more clear cluster structures; while in the case of fuzzy clusters (like in this simulation), cross-validation naturally finds lower values of ϵ. This means that regardless of the fuzziness and the number of clusters in the true model, by choosing a proper ϵ using cross-validation, CEF can achieve the closest structure to the truth. In contrast, the primary assumption for the specialized algorithms for modular networks like SIMoNe is that the clusters are well separated from each other which is not the case in this experiment. That explains the atypical shape of the cross-validation curve in Figure 1(e) for C: it cannot determine the best number of clusters for the network in Figure 1(a).

Highway Traffic Networks: We use the GGM realization of our framework to model the traffic flow networks of highways (Singliar and Hauskrecht, 2007). In particular, we apply our method to traffic flow data from highways of the Pittsburgh area gathered in 2003. The dataset consists of the measurements of 181 volume sensors ($p = 181$) gathered at different times of day for 243 workdays ($n = 243$) throughout 2003. Each measurement in the dataset is the number of vehicles passing the respective sensor during a five-minute interval. In order to generate I.I.D. samples, we chunked the original dataset into the smaller ones so that each small dataset contains the sensor measurements for a fixed time of day throughout 243 days. This way, each small dataset contains exactly 243 records and the time dependency between the records in the same dataset is minimized.

The small ratio of n/p in this problem demands some sort of regularization to learn a reliable model. In this specific problem, the geographical proximity seems to convey information regarding the prior probability of the edges. More precisely, if two sensors are geographically close to each other, intuitively it is more likely that they are directly correlated to each other or, in other words, there must be an edge between them in the underlying model. This promotes the use of PEF which enables us to encode prior belief regarding each single edge. To compute the edge priors, for each sensor we have assigned the high prior weights to the edges which connect that sensor to $k = 8$ closest sensors (based on Euclidean distance). All the other edges in the network are assigned low prior weights. These weights are fed into the algorithm through β_{ij}s in PEF.

Figure 2 shows the learned networks for 4 different times of day each mapped onto the Pittsburgh area's highway map. The results are somewhat interesting. In spite of the low prior belief due the geographical distance, during the rush hours in the morning, when there is a high and continuous inflow of traffic into the congested downtown (in the center), many of the distant sensors turn out to be directly correlated. On the other hand, during the non-rush hours,

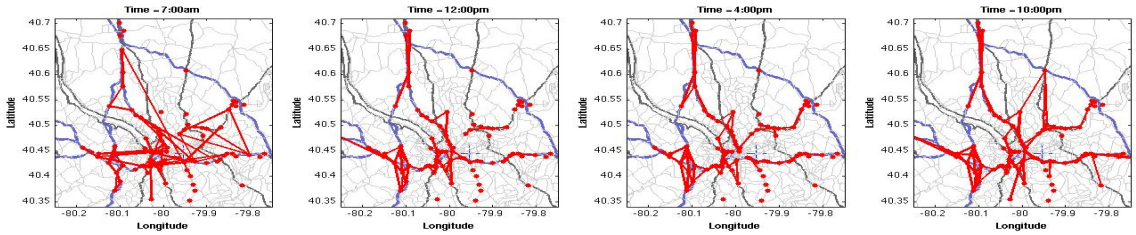

Figure 2: The learned networks at 7:00am (top-left), 12:00pm (top-right), 4:00pm (bottom-right) and 10:00pm (bottom-right)

when the traffic volumes are lower and flow freely (especially at night), the learned structure coincides more with the underlying road network (although, even in this case, there are some unexpected correlations which cannot be uncovered merely from the road map). This suggests that the geographical proximity is not the only cause of correlation, but congestion (during the rush hours in the downtown) or other events (e.g. scheduled sport event) may also cause distant sensors to be correlated.

Conclusions

We have proposed a new framework for learning the structure and parameters of pairwise Markov networks (PMNs). Unlike previous methods that bias the learning towards sparse structures, our approach is more flexible and lets one define priors over many different network structures. We have proposed a class of priors based on additive energy functions that lead to more efficient inferences and learning. We introduced, studied and tested two priors from this class. In addition to flexible structural priors, our framework permits and can learn different types of PMNs, for example, GGMs or binary PMNs. The learning algorithm that powers our framework is a variational EM method that is used to estimate both the structure and the parameters of a PMN.

We have evaluated our framework on two PMN learning problems. On a synthetic GGM data our approach was able to outperform, in terms of structure recovery, other state-of-the-art learning algorithms. On a real-world traffic network data we showed our model was able to recover solely from observations of traffic flows in different locations reasonably well the topology of the underlying highway network.

Our framework and its flexibility offer multiple new research directions, such as study of new classes of structural priors that benefit from local substructures. We also believe the proposed framework and its flexibility will let us further extend its applicability. For example, we are currently studying, in more depth, its application to models with more flexible clustered structure.

Acknowledgements

We would like to thank Tomas Singliar for helping us with the Traffic data and Christophe Ambroise and Julien Chiquet for supporting the R Package "simone" (available on CRAN). This research was supported in part by grants 1R01LM010019-01A1 and 1R01GM088224-01 from NIH. The content is solely the responsibility of the authors and does not necessarily represent the official views of the NIH.

References

Ambroise, C.; Chiquet, J.; and Matias, C. 2009. Inferring sparse Gaussian graphical models with latent structure. *Electronic Journal of Statistics*.

Banerjee, O.; Ghaoui, L. E.; and d'Aspremont, A. 2008. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *Journal of Machine Learning Research* 9:485–516.

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

Höfling, H., and Tibshirani, R. 2009. Estimation of sparse binary pairwise markov networks using pseudo-likelihoods. *J. Mach. Learn. Res.* 10:883–906.

Jiroušek, R., and Přeučil, S. 1995. On the effective implementation of the iterative proportional fitting procedure. *Comput. Stat. Data Anal.* 19(2):177–189.

Jordan, M.; Ghahramani, Z.; Jaakkola, T.; and Saul, L. 1998. An introduction to variational methods for graphical models. Technical Report CSD-98-980.

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

Lee, S.-I.; Ganapathi, V.; and Koller, D. 2007. Efficient structure learning of markov networks using l_1-regularization. In Schölkopf, B.; Platt, J.; and Hoffman, T., eds., *Advances in Neural Information Processing Systems 19*. Cambridge, MA: MIT Press. 817–824.

Marlin, B. M., and Murphy, K. P. 2009. Sparse gaussian graphical models with unknown block structure. In Danyluk, A. P.; Bottou, L.; and Littman, M. L., eds., *ICML*, volume 382 of *ACM International Conference Proceeding Series*, 89. ACM.

Meinshausen, N., and Bühlmann, P. 2006. High-dimensional graphs and variable selection with the lasso. *Annals of Statistics* 34(3):1436–1462.

Singliar, T., and Hauskrecht, M. 2007. Modeling highway traffic volumes. In *ECML*, volume 4701 of *Lecture Notes in Computer Science*, 732–739. Springer.

Wainwright, M. J.; Ravikumar, P.; and Lafferty, J. D. 2007. High-dimensional graphical model selection using ℓ_1-regularized logistic regression. In Schölkopf, B.; Platt, J.; and Hoffman, T., eds., *Advances in Neural Information Processing Systems 19*. Cambridge, MA: MIT Press. 1465–1472.