The von Mises Graphical Model: Expectation Propagation for Inference

Narges Razavian, Hetunandan Kamisetty, Christopher James Langmead

September 2011
CMU-CS-11-130
CMU-CB-11-102

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

The von Mises model encodes a multivariate circular distribution as an undirected probabilistic graphical model. Presently, the only algorithm for performing inference in the model is Gibbs sampling, which becomes inefficient for large graphs. To address this issue, we introduce an Expectation Propagation based algorithm for performing inference in the von Mises graphical model. Our approach introduces a moment-matching technique for trigonometric functions to approximate the Expectation Propagation messages efficiently. We show that our algorithm has better speed of convergence and similar accuracy compared to Gibbs sampling, on synthetic data as well as real-world data from protein structures.

This research was supported by NSF IIS-0905193. Corresponding Author: cjl@cs.cmu.edu

Keywords: Inference, Expectation Propagation, von Mises, Probabilistic Graphical Models, Proteins

1 Introduction

The von Mises distribution is a tractable continuous probability distribution that can be used to model circular variables. Recently, the undirected von Mises probabilistic graphical model (vGM) was introduced to model arbitrary collections of von Mises-distributed random variables in a compact form [9, 8]. The vGM has a number of uses in directional statistics. For example, they are increasingly used to model biomolecular structures (e.g., proteins), which can be defined in terms of a set of bond and dihedral angles.

Razavian *et al* [9, 8] have recently developed an algorithm that employs block-L_1 regularization to learn the topology and parameters of vGMs from data, and a Gibbs sampler for performing inference. Unfortunately, Gibbs sampling is not well suited to performing inference in graphical models with hundreds or thousands of variables, such as those used to model protein structures. Thus, there is a need for additional inference algorithms for vGM.

Belief Propagation (BP) [6] is a well-studied alternative to Gibbs sampling. Unfortunately, BP cannot, in general, be applied to continuous-valued distributions, unless the distributions are closed under conditioning and marginalization (e.g. Conditional Linear Gaussian Graphical Models [10]). The von Mises distribution is not closed under marginalization, and so Belief Propagation isn't applicable for vGM. However, Expectation Propagation (EP) [5] is commonly used to perform approximate inference in continuous distributions. In this paper we derive the necessary equations for an EP-style inference algorithm in vGM, and then show that the algorithm outperforms Gibbs sampling in terms of efficiency, while still retaining the accuracy level of the fully converged Gibbs sampling.

2 The von Mises Graphical Model (vGM)

Let $\Theta = (\theta_1, \theta_2, ..., \theta_p)$, where $\theta_i \in (-\pi, \pi]$. The multivariate von Mises distribution [3] with parameters $\vec{\mu}, \vec{\kappa}$, and Λ is given by:

$$f(\Theta) = \frac{\exp\left\{\vec{\kappa}^T C(\vec{\Theta}, \mu) + \frac{1}{2} S(\vec{\Theta}, \mu) \Lambda S(\vec{\Theta}, \mu)^T\right\}}{Z(\vec{\mu}, \vec{\kappa}, \Lambda)},$$

where $\vec{\mu} = [\mu_1, \mu_2, \cdots, \mu_p]$, $\vec{\kappa} = [\kappa_1, \kappa_2, \cdots, \kappa_p]$, $C(\vec{\Theta}, \mu) = [\cos(\theta_1 - \mu_1), \cos(\theta_2 - \mu_2), \cdots, \cos(\theta_p - \mu_p)]$, $S(\vec{\Theta}, \mu) = [\sin(\theta_1 - \mu_1), \sin(\theta_2 - \mu_2), \cdots, \sin(\theta_p - \mu_p)]$, Λ is a $p \times p$ matrix such that $\Lambda_{ii} = 0$, and $\Lambda_{ij} = \lambda_{ij} = \lambda_{ji}$, and $Z(\vec{\mu}, \vec{\kappa}, \Lambda)$ is the normalization constant.

Figure 1 shows the factor graph representation of the graphical mode for four variables. Under this representation the node potentials (i.e. univariate factors) are defined as $f_i = \kappa_i cos(\theta_i - \mu_i)$ and the edge potentials (i.e. bivariate factors) are defined as $f_{ij} = \lambda_{ij} sin(\theta_i - \mu_i) sin(\theta_j - \mu_j)$. Like all factor graphs, the model encodes the joint distribution as the normalized product of all factors:

$$P(\Theta = \theta) = \frac{1}{Z} \prod_{a \in A} f_a(\theta_{ne(a)}),$$

where A is the set of factors and $\theta_{ne(a)}$ are the neighbors of f_a (factor a) in the factor graph.

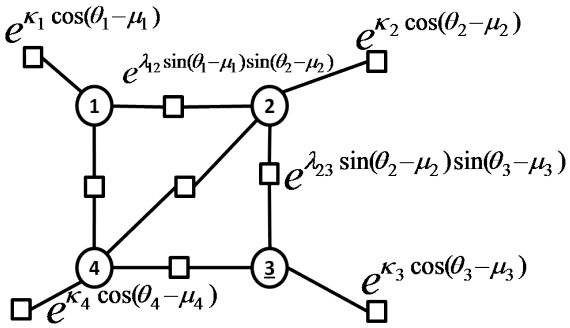


Figure 1: Factor Graph Representation for multivariate von Mises distribution. Each circular node is a variable, and the square nodes are factors.

3 Inference in von Mises Graphical Models

One of the most widely used inference techniques on graphical models is Belief Propagation [6]. The Belief Propagation algorithm works by passing real valued functions, called $messages$, along the edges between the nodes. A message from node v to node u contains the probability factor which results from eliminating all other variables up to u, to the variable u.

There are two types of messages: (i) A message from a variable node v to a factor node u, which is the product of the messages from all other neighboring factor nodes:

$$\mu_{v\to u}(x_u) = \prod_{u^*\in N(v)\setminus\{u\}} \mu_{u^*\to v}(x_v),$$

where $N(v)$ is the set of neighboring (factor) nodes to v. (ii) A message from a factor node u to a variable node v is the product of the factor with messages from all other nodes, marginalized over all variables except x_v:

$$\mu_{u\to v}(x_v) = \int_{\mathbf{x}'_u : x'_v = x_v} f_u(\mathbf{x}'_u) \prod_{v^*\in N(u)\setminus\{v\}} \mu_{v^*\to u}(x'_u) dx'_u,$$

where $N(u)$ is the set of neighboring (variable) nodes to u.

In von Mises graphical models, the integration step for computing the message from a bivariate factor to a node does not have a closed form solution, so that message can not be represented by a single set of μ and κ and λ anymore, and thus we can not perform belief propagation on a vGM directly.

A solution to this problem is to approximate these integrals using a chosen distribution family, and only propagate the parameters of the messages. This method is called Expectation Propagation [5].

3.1 Expectation propagation

Expectation Propagation is similar to Belief Propagation, except that the messages that are exchanged between the nodes only contain the $expectation$ of the marginalization, rather than the

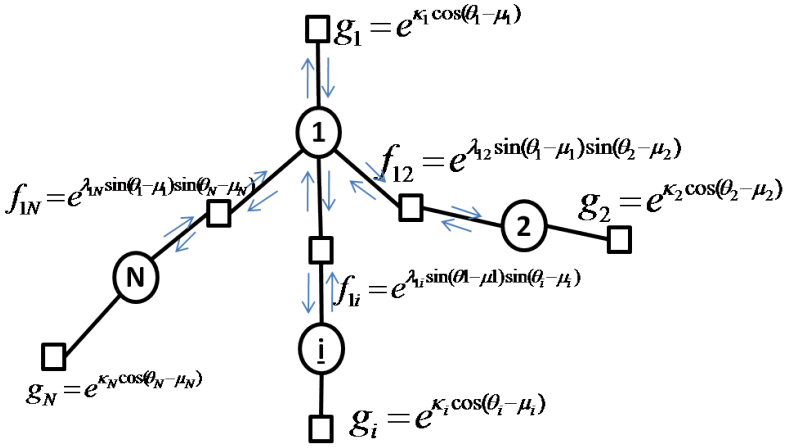


Figure 2: Von Mises Factor graph representation, and the messages exchanged between the factors and variable x_1.

exact values [5]. Given a factorized probability distribution $P(x) = \prod_{i=1}^{N} t_i(x)$, expectation propagation computes an approximated probability $q(x) = \prod_{i=1}^{N} \tilde{t}_i(x)$ such that the KL-divergence, $KL(p(x), q(x))$, is minimized.

In the next section we will perform the moment matching step (i.e. The KL-divergence minimization step) for the special case of the von Mises graphical model.

3.2 Expectation propagation for von Mises Graphical Models

Recall from section 2, that a von Mises distribution is factorized as uni- and bivariate factors:

$$P(x|\mu, \kappa, \lambda) = \frac{1}{Z(\kappa, \lambda)} \prod_{i=1}^{n} e^{\kappa_i \cos(x_i - \mu_i)} \prod_{i,j=1 i \neq j}^{n} e^{\lambda_{ij} \sin(x_i - \mu_i) \sin(x_j - \mu_j)}$$

Figure 2 shows the factor graph representation of von Mises and the messages that pass between variable x_1 and the related factors.

As mentioned in previous section, in Expectation Propagation, messages contain the expectations rather than the actual parameters. In EP for von Mises, each message going to the variable will be approximated by a $univariate$ von Mises and we use moment matching to calculate the messages.

There are four types of messages: (i) Messages from (univariate) factor node g_i to the variable x_i, and (ii) vice versa; (iii) messages from (bivariate) factor node f_{ij} to the variable x_i, and (iv) vice versa.

Messages from factor g to the variables - Factors g_i are univariate factors with $e^{\kappa_i cos(x_i - \mu_i)}$ as their function. The message they will send to the variable x_i is simply $e^{\kappa_i cos(x_i - \mu_i)}$, which is already in the desired form of a univariate von Mises function.

Messages from factor f to the variables - Factors f_{ij} are bivariate, and they first receive message from x_j, then multiply their potential (a function of the form $e^{\lambda_{ij} sin(x_i-\mu_i)sin(x_j-\mu_j)}$) to the incoming message from x_j, and then approximate the integration by a message of the form $e^{\kappa^* cos(x_i-\mu^*)}$.

Let's assume that the message $m_{j\to f_{ij}} = e^{\kappa_j^* cos(x_j-\mu_j^*)}$ is the message that x_j sends to the factor f_{ij}. The message that f_{ij} sends to the variable x_i is then calculated as:

$$m_{fij\to x_i}(x_i) = \int_{x_j} f_{ij}(x_i, x_j) m_{j\to f_{ij}}(x_j) dx_j = \int_{x_j} e^{\lambda_{ij} sin(x_i-\mu_i)sin(x_j-\mu_j)} e^{\kappa_j^* cos(x_j-\mu_j^*)} dx_j$$

We want to approximate this integral by the form:

$$m_{fij\to x_i}(x_i) \approx e^{\kappa^* cos(x_i-\mu^*)}$$

We will use moment matching techniques to find κ^* and μ^*. Unfortunately, direct moment matching does not have a closed form solution in this case and is computationally very expensive. Instead, we use *Trigonometric Moments*, to perform moment matching and approximate the message. [4] defines two set of moments, $\alpha_0, \alpha_1, ...$ and $\beta_0, \beta_1, ...$, that are used to identify a function with desired level of accuracy. These moments are defined as:

$$\alpha_p(f_x) = \int_x \cos(px) f(x) dx$$
$$\beta_p(f_x) = \int_x \sin(px) f(x) dx$$

Note that the series α and β specify the $Fourier$ coefficients of a function, which uniquely specifies the function with desired accuracy. We match α_0 , α_1 , β_0 and β_1 of these two functions: The actual messsage that should be sent, $\int_{x_j} e^{\lambda_{ij} sin(x_i-\mu_i)sin(x_j-\mu_j)} e^{\kappa_j^* cos(x_j-\mu_j^*)} dx_j$, and the approximation of it, $e^{\kappa^* cos(x_i-\mu^*)}$.

For simplicity, and without loss of generality, we can assume that the μ_i for data was first calculated and subtracted from the data. Therefore we have:

$$\alpha_0^{real} = \iint_{x_i,x_j=-\pi}^{\pi} e^{\lambda_{ij} sin(x_i)sin(x_j)+\kappa_j^* cos(x_j-\mu_j^*)} dx_j dx_i$$
$$\beta_0^{real} = 0.$$
$$\alpha_1^{real} = \iint_{x_i,x_j=-\pi}^{\pi} \cos(x_i) e^{\lambda_{ij} sin(x_i)sin(x_j)+\kappa_j^* cos(x_j-\mu_j^*)} dx_j dx_i$$
$$\beta_1^{real} = \iint_{x_i,x_j=-\pi}^{\pi} \sin(x_i) e^{\lambda_{ij} sin(x_i)sin(x_j)+\kappa_j^* cos(x_j-\mu_j^*)} dx_j dx_i$$

The trigonometric moments for $e^{\kappa^* cos(x_i-\mu^*)}$ are also calculated:

$$\alpha_0^{ep} = \int_{x_i=-\pi}^{\pi} e^{\kappa^* cos(x_i-\mu^*)} dx_i = 2\pi I_0(\kappa^*)$$
$$\beta_0^{ep} = 0$$

On the other hand, for the higher degree moments we have:

$$\begin{aligned}
\alpha_1^{ep} &= \int_{x_i-\pi}^{\pi} \cos(x_i) e^{\kappa^* \cos(x_i-\mu^*)} dx_i \\
&= \int_{x_i-\pi}^{\pi} \cos(x_i) e^{\kappa^* \cos(x_i)\cos(\mu^*)-\kappa^* sin(x_i) sin(\mu^*)} dx_i \\
\beta_1^{ep} &= \int_{x_i-\pi}^{\pi} \sin(x_i) e^{\kappa^* cos(x_i-\mu^*)} dx_i \\
&= \int_{x_i-\pi}^{\pi} \sin(x_i) e^{\kappa^* \cos(x_i)\cos(\mu^*)-\kappa^* sin(x_i) sin(\mu^*)} dx_i
\end{aligned}$$

Each individual integral does not have a closed form on its own, and thus would not let us estimate the κ^* and μ^*. However we can combine the α_1^{ep} and β_1^{ep}s, and get the relationship between κ^*, μ^*, α_1^{real} and β_1^{real}:

$$\kappa^* \sin(\mu^*)\beta_1^{ep} + \kappa^* \cos(\mu^*)\alpha_1^{ep} = e^{\kappa^* \cos(x_i-\mu^*)}\Big|_{x_i=-\pi}^{\pi} = 0$$

Thus, after moment matching $\alpha_p^{real} = \alpha_p^{ep}$ and $\beta_p^{real} = \beta_p^{ep}$, and so we obtain the parameters of the Expectation Propagation message:

$$\kappa^* = \frac{I_0^{-1}(\alpha_0^{real})}{2\pi} \;\; and \;\; \mu^* = -tan^{-1}(\frac{\alpha_1^{real}}{\beta_1^{real}})$$

Messages from variable x_i to the factors - Messages sent from factors are all approximated to be of the form $e^{\kappa_j^* cos(x_i-\mu_j^*)}$. The messages that variable sends to the factors are also of the same family, however they are exact. A message from variable x_i to factor g_i is a product of all messages received excluding the message received from factor g_i:

$$\begin{aligned}
m_{i\to g_i}(x_i) &= \prod_{j=1..N, j\neq i} m_{f_{ij}\to i}(x_i) \\
&= \prod_{j=1..N, j\neq i} e^{\kappa_j^* \cos(x_i-\mu_j^*)}
\end{aligned}$$

$$m_{i\to g_i}(x_i) = e^{\kappa_i^G \cos(x_i-\mu_i^G)}$$

Thus, the message parameters κ_i^G and μ_i^G are:

$$\begin{aligned}
\kappa_i^G &= \sqrt{(\sum_{l\neq i} \kappa_l^* \cos(\mu_l^*))^2 + (\sum_{l\neq i} \kappa_l^* \sin(\mu_l^*))^2} \\
\mu_i^G &= \tan^{-1} \frac{\sum_{l\neq i} \kappa_l^* \sin(\mu_l^*)}{\sum_{l\neq i} \kappa_l^* \cos(\mu_l^*)}
\end{aligned}$$

Similarly, messages from variable x_i to factor f_{ij} are:

$$m_{i \to f_{ij}}(x_i) = e^{\kappa^F_{ij} \cos(x_i - \mu^F_{ij})},$$

such that

$$\kappa^F_{ij} = \sqrt{(\sum_{l \neq j} \kappa^*_l \cos(\mu^*_l))^2 + (\sum_{l \neq j} \kappa^*_l \sin(\mu^*_l))^2}$$
$$\mu^F_{ij} = \tan^{-1} \frac{\sum_{l \neq j} \kappa^*_l \sin(\mu^*_l)}{\sum_{l \neq j} \kappa^*_l \cos(\mu^*_l)}$$

Once these messages are computed, we calculate the final partition function as the integration of beliefs of any of the variables.

$$\begin{aligned} Z &= \int_{x_i} \prod_{j=1..N} e^{\kappa^*_j \cos(x_i - \mu^*_j)} dx_i \\ &= \int_{x_i} e^{\sum_{j=1..N} \kappa^*_j \cos(x_i - \mu^*_j)} dx_i \\ &= \int_{x_i} e^{\kappa^*_z \cos(x_i - \mu^*_z)} dx_i \end{aligned}$$

$$\text{where } \kappa^*_z = \sqrt{(\sum_{l=1..N} \kappa^*_l \cos(\mu^*_l))^2 + (\sum_{l=1..N} \kappa^*_l \sin(\mu^*_l))^2}$$
$$\mu^*_z = \tan^{-1} \frac{\sum_{l=1..N} \kappa^*_l \sin(\mu^*_l)}{\sum_{l=1..N} \kappa^*_l \cos(\mu^*_l)}$$

Finally, the partition function is calculated as $Z = \frac{1}{2\pi} I_0(\kappa^*_z)$.

4 Experiments

We implemented the Expectation Propagation (EP) inference algorithm for vGM using the functions derived in previous section. In this section, we show that the EP algorithm is much faster than Gibbs at a minimal loss in accuracy thus presenting a viable alternative in large graphs. We also compare EP and Gibbs sampling's accuracy on both synthetic and real data, and show that the two method give similar results in terms of the imputation error.

4.1 Synthetic Data and Gibbs Sampling for vGM

The synthetic data for our tests were generated by first randomly generating von Mises graphical models of size 8, 16, 32, and 64 with varying edge density, and then performing Gibbs sampling

on those models. According to [2], von Mises conditionals are univariate von Mises distributions, so Gibbs sampling can be performed easily. In particular:

$$f(x_p|x_1, x_2, \ldots x_{p-1}) \propto \exp\{\kappa_p cos(x_p - \mu_p) + \sum_{j=1}^{p-1} \lambda_{jp} sin(x_j - \mu_j) sin(x_p - \mu_p)\}$$
$$= \exp\{\kappa^* cos(x_p - \mu^*)\}$$

$$\text{where, } \kappa^* = \sqrt{\kappa_p^2 + (\sum_{j=1}^{p-1} \lambda_{jp} sin(x_j - \mu_j))^2}$$
$$\mu^* = \arctan(\frac{1}{\kappa_p} \sum_{j=1}^{p-1} \lambda_{jp} sin(x_j - \mu_j))$$

In our sampler, the values of κ were randomly drown from a uniform distribution between $[0..S_k]$ with S_k set to 1; and λ values were drown from a Normal distribution $N(0, S_\lambda)$ where S_λ was 1.

The sampler is initiated with a completely random vector, and we sample each variable conditioned on the values of the rest of the variables, based on the probabilities calculated above. Initially the samples are not from a valid von Mises distribution, however after enough samples are visited and the sampler has $converged$, we collect the next set of generated samples as valid vGM samples.

4.2 Expectation Propagation vs. Gibbs Sampling Experiment, over Synthetic Dataset

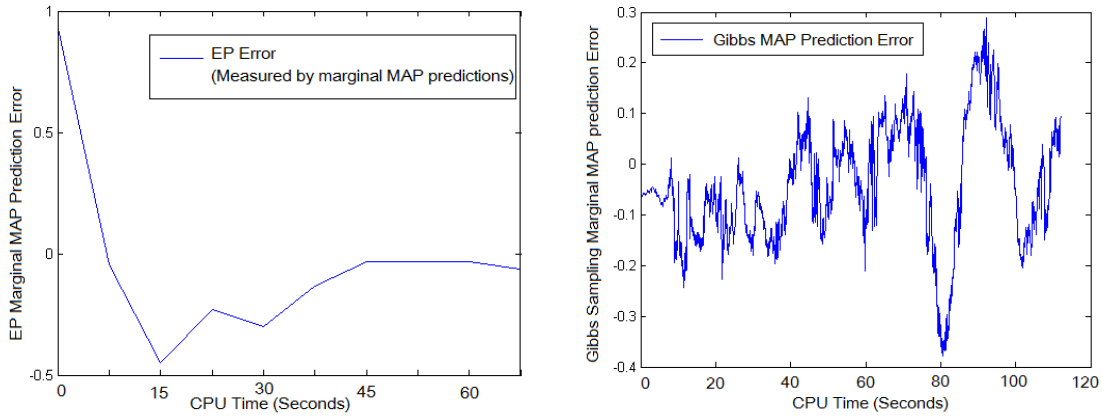
Table 1 shows the imputation error for EP and Gibbs sampling method, over datasets of size 3000 samples, generated for 8, 16, 32 and and 64-node vGM graphical models. The graph density was 10% for all cases, and the values for S_k and S_λ are both equal to 1. To compute the imputation error, for each sample in the dataset and for each variable of that sample, we computed the most likely value for that variable, conditioned on the values of all other variables. For each algorithm, we then computed the circular mean [4] of the distances between the predicted value and the actual value of the sample variable, averaged over all 3000 the samples.

Table 1: Mean imputation error for Expectation Propagation and Gibbs sampling on the synthetic dataset with 10% graph density. (Errors in radians)

Imputation Error	8 Nodes	16 Nodes	32 Nodes	64 Nodes
Expectation Propagation	-0.4639	-0.0074	0.0500	-0.0600
Gibbs Sampling	0.4161	0.0039	-0.0073	0.0172

As can be seen, the error of EP is only slightly higher than the Gibbs sampling error. This result matches our expectation, since EP is an approximate algorithm and in the best case, it should get the same accuracy as the fully converged Gibbs sampling. We can see that the divergence of accuracy is quite small.

The EP inference algorithm, however, is much more efficient than the Gibbs sampler (as expected). Figure 3 shows the progress through time, of Gibbs sampling versus Expectation Propagation on the synthetic von Mises graphical model of size 16, with 10% graph density. We measured this progress by computing the MAP estimate of all 16 variables through time, by each of the two algorithms, and then we computed the average divergence of this estimate from the mean of the samples. We observe that for a fixed budget of time, EP gives more reliable results. Figure 4 shows the total CPU time that it takes for EP inference versus Gibbs sampling for up to 128 variables. As the size of the graph increases, we see that EP gains a larger advantage in terms of convergence time, over Gibbs sampling (as expected).



(a) (b)

Figure 3: The convergence analysis for Expectation Propagation and Gibbs sampling.

4.3 Expectation Propagation vs. Gibbs Sampling Experiment, over Ubiquitin Protein Data

The three dimensional structure of a protein (and other molecules) can be defined in terms of a set of atomic coordinates or, equivalently, in terms of a set of dihedral angles describing the backbone and side-chain configurations of each amino acid in the protein. We used an algorithm presented in [9, 8] to learn a regularized structure and parameters of a VGM from data, to learn a VGM model of the distribution of torsion angles for the protein Ubiquitin. Our training data were obtained via Molecular Dynamics (MD) simulation, which simulates the dynamics of the protein's structure by integrating Newton's laws of motion. As is customary, a new protein structure was written to disk after every 10,000 integration steps. Thus, informally, MD produces a set of samples from

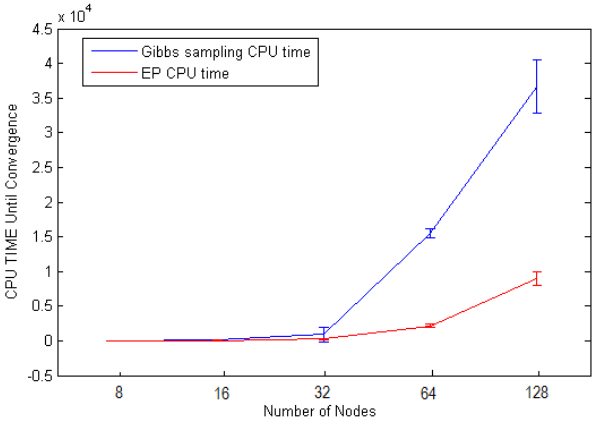
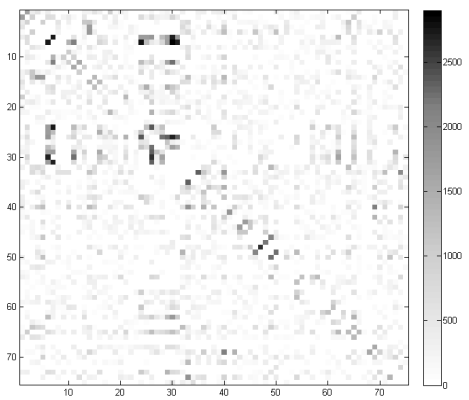
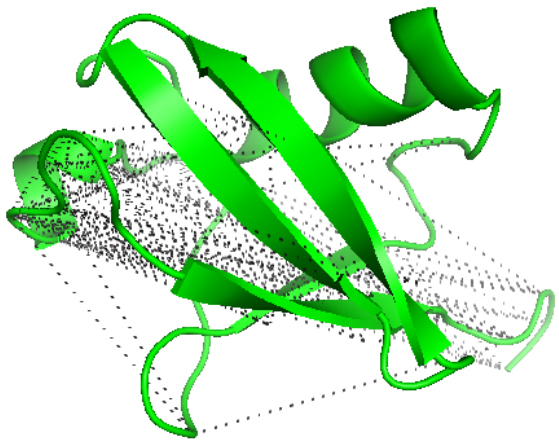


Figure 4: The convergence rate for Expectation Propagation and Gibbs sampling (CPU Time measured in seconds.)

an underlying distribution over structures. Our Ubiquitin MD data includes 15,000 conformations, each represented as a vector of 225 torsion angles that define the structure of the protein. Figure 5 shows the graph structure learned on this Ubiquitin data.



(a) The estimated dependency parameters for vGM learned over Ubiquitin data, projected to protein residue numbers.



(b) The learned von Mises graphical model edges projected on the Ubiquitin protein structure.

Figure 5: von Mises graphical model learned over Ubiquitin data.

Having learned the vGM from the MD data, we then computed the imputation errors for EP and Gibbs sampling (Table 2) over all the 15,000 samples. Starting from random parameters, the EP algorithm converged in approximately 5 hours. Gibbs sampling did not converge in our time limit. The average imputation error is displayed in the table.

Table 2: Imputation Error for Expectation Propagation and Gibbs sampling on Ubiquitin protein torsion angles

	Expectation Propagation	Gibbs Sampling
Imputation Error	-0.0857	-0.0404

From the results in Table 2, we can see that the EP method has slightly worse imputation error compared to Gibbs sampling, however both errors are small. Our result is in agreement with other EP experiments[5], [1], [7] where the EP performance is close to the fully converged Gibbs Sampling. The benefit of EP is in the increased efficiency, which makes the inference on larger vGM graphs possible.

5 Conclusion and Future Work

In this paper we presented a novel algorithm for inference in von Mises graphical models. Our algorithm uses Expectation Propagation on the graph. This problem is non-trivial because it requires moment matching on circularly distributed random variables. We solved that sub-problem by employing trigonometric moment matching to derive the estimated parameters for the the EP messages. Our algorithm was then demonstrated to be superior to Gibbs sampling in terms of computational efficiency, while achieving comparable accuracies.

This work was motivated by the desire to model protein and other biomolecular structures. Protein structures can be defined in terms of (typically) hundreds to thousands of dihedral angles. Thus, scalability is critical. Our EP-based inference algorithm is relevant to a number of important tasks, such as protein structure prediction and, more generally, to modeling the conformational variability in biological macromolecules induced by environmental changes (e.g., binding to another molecule).

One important limitation of our approach is that the distribution of each variable is assumed to be unimodal. Many important applications require multi-modal distributions over circular variables. Thus, as part of ongoing work we are investigating the extension of vGM to multi-modal distributions. However, inference and learning for multi-modal continuous-valued distributions is known to be difficult, and will likely require the adoption of non-parametric techniques. We are actively exploring such techniques.

References

[1] John Guiver and Edward Snelson. Bayesian inference for plackett-luce ranking models. Technical report, 2009.

[2] Gareth Heughes. *Multivariate and time series models for circular data with applications to protein conformational angles*. PhD Thesis, Department of Statistics, University of Leeds.

[3] K. V. Mardia. Statistics of directional data. *J. Royal Statistical Society. Series B*, 37(3):349–393, 1975.

[4] K.V. Mardia and P.E. Jupp. *Directional statistics*. Wiley Chichester, 2000.

[5] Thomas P. Minka. Expectation propagation for approximate bayesian inference. In *Uncertainty in Artificial Intelligence*, pages 362–369, 2001.

[6] Judea Pearl. Fusion, propagation, and structuring in belief networks. *Artif. Intell.*, 29(3):241–288, 1986.

[7] Yuan Qi. Extending expectation propagation for graphical models, 2004.

[8] Narges Sharif Razavian, Hetunandan Kamisetty, and Christopher James Langmead. The von mises graphical model: Regularized structure and parameter learning. Technical Report CMU-CS-11-129, Carnegie Mellon University, 2011.

[9] Narges Sharif Razavian, Hetunandan Kamisetty, and Christopher James Langmead. The von mises graphical model:structure learning. Technical Report CMU-CS-11-108, Carnegie Mellon University, 2011.

[10] Sam Roweis and Zoubin Ghahramani. A unifying review of linear gaussian models, 1999.