VARIATIONAL INFERENCE METHODS FOR CONTINUOUS PROBABILISTIC GRAPHICAL MODELS

by

Yuanzhen Guo

APPROVED BY SUPERVISORY COMMITTEE:

Nicholas Ruozzi, Chair

Vibhav Gogate

Sriraam Natarajan

Vincent Ng

To mom, dad, grandma and grandpa

VARIATIONAL INFERENCE METHODS FOR CONTINUOUS PROBABILISTIC GRAPHICAL MODELS

by

YUANZHEN GUO, BS, MS

DISSERTATION

Presented to the Faculty of

The University of Texas at Dallas

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY IN

COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT DALLAS

December 2020

ACKNOWLEDGMENTS

I would like to give my biggest thanks to my supervisor Nicholas Ruozzi who introduced me to graphical models and guided me throughout my Ph.D. with wisdom and patience. He is and will always be my super star in research.

I would like to thank my committee members Dr. Vibhav Gogate, Dr. Sriraam Natarajan and Dr. Vincent Ng for their time and efforts to evaluate my dissertation and for their valuable advises on my research.

I also would like to thank my friends and colleagues: David, Li, Tahrima, Chiradeep and Shasha. I really learned a lot from our discussions and we had the best lab in the world.

Finally, I would like to thank my family for their love and support all the way along. They deserve more than half of my medals.

November 2020

VARIATIONAL INFERENCE METHODS FOR CONTINUOUS PROBABILISTIC GRAPHICAL MODELS

Yuanzhen Guo, PhD
The University of Texas at Dallas, 2020

Supervising Professor: Nicholas Ruozzi, Chair

Graphical models provide a general framework for representing and reasoning about data. Once these models are fit to data, they can be used to answer statistical queries about the observed data. Unfortunately these answering these queries, or performing inference, is NP-hard in general. To tackle this problem, many approximate inference methods have been proposed. Belief propagation, a widely used inference method for probabilistic graphical models, is exact on tree-structured models but does not guarantee convergence or correctness on general graphs. Alternative approaches based on variational inference have also been proposed. Variational inference methods start by approximating the intractable model with a more friendly surrogate model by minimizing the KL divergence between the surrogate and original models. The marginals from the surrogate model can then be treated as approximations to the intractable marginals of original model. However the performance of variational inference methods can be highly dependent on the surrogate model and performance can be terrible if surrogate model cannot well approximate the original model. Another drawback of variational inference methods is that extra approximations are necessary in some cases to make the approach computationally feasible. Also it is very hard to prove any theoretical guarantees for these methods.

In this work, we propose a new variational inference method that adopts a mixture of independent distributions as our surrogate model. Instead of minimizing the KL divergence between surrogate and original model, we propose to maximize the Bethe free energy with respect to the surrogate marginals standard optimization strategies. Our method has many advantages compared to existing methods and is provably correct under certain conditions. We demonstrate the superior performance of our method on a variety of large-scale real-world problems, where we show that not only can our method can achieve better results on these tasks compared existing state-of-the-art methods, but it can also be implemented efficiently at scale.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Probabilistic graphical models (PGMs) such as Bayesian networks and Markov random fields provide a unified framework for modeling and prediction. They have been applied in diverse fields such as machine learning, bioinformatics, natural language processing, computer vision, image processing, and more (Koller and Friedman, 2009; Wainwright and Jordan, 2008; Blei et al., 2003; Fleet and Weiss, 2006a). In these models, dependencies among random variables are encoded via a graphical structure, which can then be exploited to build efficient exact and approximate inference routines. In this way, PGMs bring together the study of graph theory, probability theory, and algorithms.

The concept of PGMs was first introduced by Pearl and Paz (1985) in the mid 80s. Bayesian networks (also known as belief networks), introduced by Pearl (1985), are a family of probabilistic graphical models defined on a directed acyclic graphs (DAGs), where vertices represent the random variables and directed edges capture statistical dependencies.Markov random fields (Ross Kindermann, 1980) are a family of PGMs defined on a undirected factor graph, where each factor is represented by a function (potential) defined over a clique (a single node, an edge, a triangle, etc.). Similar to Bayesian networks, MRFs also can represent the dependencies between variables, but MRFs can have cycles in their graphical structure, which allows MRFs the ability to model certain dependencies that Bayesian networks cannot. These PGMs and their variants, such as clique trees (Halin, 1976), hidden Markov models (HMM) (Lafferty et al., 2001), conditional random fields (CRF) (Stratonovich, 1965), etc., have found applications in a wide variety of problem domains.

Probabilistic graphical models are usually fit to data, e.g., using maximum likelihood estimation, and then reasoning/prediction (inference) is performed on the learned models. A variety of different inference queries can be performed. In this work, we will be primarily concerned with marginal inference, i.e., computing marginal distributions over subsets of the

random variables, maximum a posteriori (MAP) inference, i.e., finding an assignment of the random variables that maximizes the joint probability, and marginal MAP inference, which combines the two. Unfortunately both marginal inference and MAP inference are NP-hard (Cooper, 1990; Park, 2002), especially when the number of random variables in the model is large.

Since exact inference in these models is intractable in general, approximate inference algorithms, which sacrifice accuracy in exchange for scalability, have been popular in practice. The belief propagation (BP) algorithm (Pearl, 1982), originally developed as an exact inference algorithm on tree-structured graphs, can be extended to yield an approximate inference method on general PGMs. These kinds of approximate message-passing algorithms are most useful when the random variables under consideration are from a discrete domain. When the random variables are continuous, the standard belief propagation algorithm can be difficult to apply: calculating the exact messages cannot be done in closed form. This necessitates additional approximations that can makes the BP method approximate even for tree-structured models. Variants of BP method have been proposed to address these issues in the continuous case, e.g. nonparametric belief propagation (NBP) (Sudderth et al., 2003), particle belief propagation (PBP) (Ihler and McAllester, 2009), expectation particle belief propagation (EPBP) (Lienart et al., 2015) and kernel belief propagation (KBP) (Song et al., 2011). However, each these approaches have their own drawbacks and limitations. First, in many of the methods, the potential functions that describe the model are often restricted, sometimes severely, in order to make the approach tractable or mathematically sound. Second, convergence guarantees can be difficult to establish for these models as they often rely on sampling in order to approximate integrals required as part of the message passing. Last but not least, the message update procedure of these methods can be either time or space consuming - especially if accurate answers are required. This limits their applicability to small-scale problems.

Variational inference methods are another popular family of approximate inference methods for PGMs Jordan et al. (1999). Variational inference (VI) approximates the joint probability distribution with a surrogate model from a more tractable family of distributions. Optimization is done to minimize the difference, typically the KL divergence, between the original model and the surrogate model. Inference queries are answered using the simpler surrogate model. The simplest VI methods are based on mean-field lower bounds (Anderson and Peterson, 1987), but more modern approaches are based on mixtures of Gaussians, e.g., nonparametric variational inference (NVI) (Gershman et al., 2012) or Stein variational methods (Wang et al., 2018). The quality of the approximation highly depends on how close the surrogate model is to the original model. In practice, the optimization procedure used to fit the tractable family is most efficient when closed form updates are possible. If the updates are not in closed form, further approximations must be made. Finally, the VI optimization problem is non-convex and NP-hard in general - the procedure need not converge to the best model from the tractable family.

In this dissertation, we propose a general purpose scalable approximate inference algorithm for PGMs. Specifically, we propose a new approximate variational that can be applied to general graphs, i.e., graphs with loops, and has no restrictions on the form of the potential functions. Out method can handle discrete variables, continuous variables, and the mixed case with both discrete and continuous variables. Our method can be applied to large-scale real world problems by taking advantage of modern GPUs. We demonstrate the efficiency and scalability of our method in several synthetic and real world inference applications; we show that our method advances the state-of-the-art by achieving better or comparable results when compared with the existing approximate inference methods discussed above.

1.1 Contributions

The main contribution of this dissertation is a novel approximate variational inference method called Bethe variational inference (BVI) method. Our method has many advantages over existing methods, including, but not limited to the following.

- Our method makes no assumptions on the potential functions in the probabilistic graphical models, other than their product must be normalizable. As such, it can handle the kinds of generic potential functions that arise in real-world applications.

- Unlike other approximations built off of the Bethe free energy, our method yields a joint distribution from which approximate joint marginals can easily be computed.

- Our method can be easily extended to the hybrid case, i.e., models with both discrete and continuous variables, with only minor modifications. This gives the flexibility of adding hidden variables (discrete or continuous) to the model.

- The gradients required for our optimization procedure can be naturally vectorized to take advantage of modern GPU/parallel computing.

- We can provide theoretical guarantees on the accuracy our method in certain cases, e.g., when using Gaussian quadrature methods for integral approximation.

- Any other expectation approximation methods can be integrated into our framework, e.g., approximate sampling methods and Stein variational methods.

- Our method often converges much faster than particle-based message-passing algorithms and frequently achieves comparable or better results.

1.2 dissertation Outline

The remaining chapters in this dissertation are organized as follows: we begin by reviewing relevant background material in Chapter 2. We introduce the definition and representation of probabilistic graphical model as well as two major types of PGMs and discuss variable elimination. Chapter 3 describes approximate message passing algorithms. We start with the belief propagation algorithm and describe several variants specifically applicable in the continuous case. Chapter 4introduces variational inference methods. In Chapter 5, we present the details of our method - Bethe variational inference. We describe the schema of our method and provide a detailed analysis. In Chapter 6, we generalize our method to mixtures of tree-structured models. Chapter 7 considers the approximating the beliefs using different types of mixture distributions, e.g., Dirichlet mixtures. Chapter 8 provides an experimental evaluation of our method on both synthetic problems and real world applications. We show that our method outperforms the state-of-the-art methods on most of the tasks. Finally, Chapter 9 concludes the dissertation and discusses both the limitations of this work and potential directions for future work.

CHAPTER 2

BACKGROUND

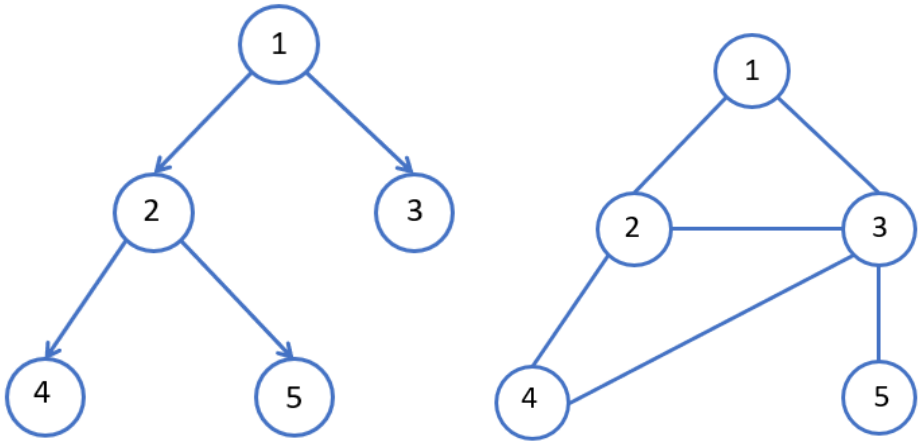
In this chapter, we introduce probabilistic graphical models (PGMs) and two popular PGMs, i.e., Bayesian networks and Markov random fields (MRFs). In addition, we describe the types of inference tasks that we will be interested in and discuss simple algorithmic approaches for exact inference.

2.1 Probabilistic Graphical Models

A graph $G = (V, E)$ is a tuple where V is a set of vertices (nodes) and $E \subseteq V \times V$ is a set of edges connecting those vertices. Graphs can be either directed or undirected. In directed graphs, $(i, j) \in E$ indicates that there is an directed edge from node i to node j. A directed graph is acyclic if there is no sequence of directed edges starting at a vertex $i \in V$ and ending at i. Given an edge $(i, j) \in E$ in a directed acyclic graph (DAG), node i is called the parent of node j and node j is the child of node i.

In the undirected case, $(i, j) \in E$ represents an undirected edge between node i and node j. An example of directed and undirected graphs can be found in Figures 2.1a and 2.1b. In an undirected graph, two nodes in the vertex set are called neighbors if they are connected by an edge in the graph. We will write $N(i)$ to denote the set of neighbors of node $i \in V$. A undirected graph is called a tree if there are no cycles, i.e., a sequence of vertices $v_0, \ldots, v_n \in V$ starting at a vertex $v_0 = i \in V$ and ending at $v_n = i \in V$ such that $(v_t, v_{t+1}) \in E$ for all $t \in \{0, \ldots, n-1\}$ and all vertices are distinct (except for v_0 and v_n).

A probabilistic graphical model (Koller and Friedman, 2009; Pearl, 2014) is described by four elements: a graph $G = (V, E)$, a set of random variables X, a domain $\mathbb{X}$ for X, and a set of functions $f \in \mathbb{F}$ that describe a joint probability distribution that *factorizes* over the graphical structure. PGMs provide a simple framework for modeling and compactly



(a) Example of a directed graph. (b) Example of an undirected graph.

Figure 2.1: Example graphs.

representing complicated distributions. In addition, the graphical structure of PGMs can be leveraged in the design of exact or approximate inference schemes in order to reduce the computational cost of these procedures. In the following subsections, we will introduce two types of most popular PGMs: Bayesian networks and Markov random fields.

2.1.1 Bayesian Networks

Consider modeling the join probability distribution of n random variables $x_1, \ldots, x_n$. Even in the simplest case with n binary random variables, i.e., $x_i \in \{0, 1\}$, the joint distribution requires the specification of $2^n - 1$ values to be completely specified. As n grows, even storing general distributions quickly becomes intractable and more compact representations of the joint distribution are required. Observe that the complexity of modeling the joint distribution would be greatly reduced if it could be factorized into product of smaller (conditional) distributions. For example, consider a joint probability distribution $P(A, B, C)$ over three random variables $\{A, B, C\}$ with domain $\{1, \ldots, k\}$, which requires the specification of $k^3 - 1$ values. However, if the joint distribution can be factorized as a product of conditional

distributions, say $P(A, B, C) = P(A)P(B|A)P(C|B)$, then $k - 1 + 2k(k - 1)$ values would be sufficient.

The above approach, formalized as Bayesian networks (BNs), was proposed by Pearl (1985). A BN is a type of PGM that represents the joint probability distribution of a collection of random variables as a product of conditional distributions associated with a directed graph. Like general PGMs, a BN consists of four elements:a directed acyclic graph (DAG) $G = (V, E)$, a set of random variables X (one for each $i \in V$), a set of domains $\mathbb{X}$ for the random variables X, and a collection of conditional probability distributions of the form $p_i(x_i|x_{parent(i)})$ for each node $i \in V$, i.e., the conditional probability of node i given its parent nodes, $parent(i)$, in the DAG. The joint probability distribution of a Bayesian network can be expressed as follows.

$$P(X) = \prod_{i \in V} p_i(x_i|x_{parent(i)}) \tag{2.1}$$

An example of a Bayesian network is shown in Figure 2.2 - note that $parent(i)$ can be the empty set.

By making the assumption that the variable's distribution of each node only depend on its parent node, Bayesian networks represent a special factorization of the joint probability distribution thus significantly reduce the model complexity. Thus Bayesian networks are perfect for combining prior knowledge into the model (Heckerman, 2008) and easy to interpret. It is also very easy to generate data from a Bayesian network. With these advantages, Bayesian networks have been broadly applied to the areas of machine learning, speech recognition, medical diagnosis and etc. (Heckerman and Nathwani, 1992; Geiger and Heckerman, 1996; Zweig and Russell, 1998).

2.1.2 Markov Random Fields

Markov random fields (MRFs) are a family of undirected probabilistic graphical models first used in statistical physics to model physical phenomena (Ross Kindermann, 1980).

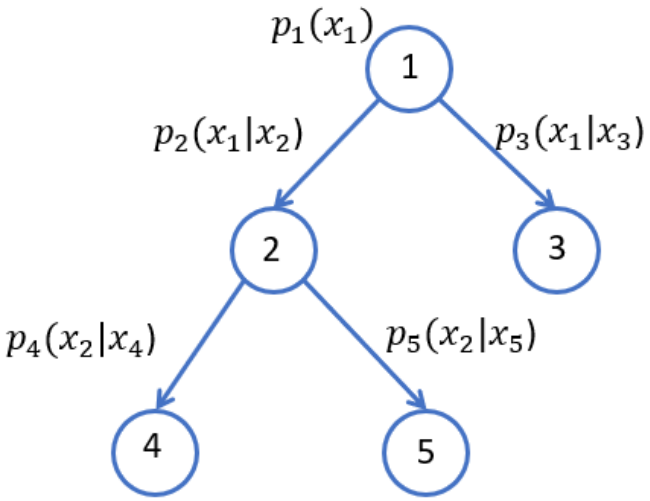


Figure 2.2: Example of Bayesian network

Given their flexibility, MRFs have been popular in the research fields of machine learning, bioinformatics, natural language processing, computer vision, and many more (Koller and Friedman, 2009; Wainwright and Jordan, 2008; Blei et al., 2003; Fleet and Weiss, 2006a; Szeliski et al., 2008).

Distinct from BNs, MRF models are defined via a collection of nonnegative potential functions defined over the cliques of an undirected graph $G = (V, E)$. In graph theory, a clique $c \subseteq V$ is a fully connected subgraph: a subset of nodes such that every pair of nodes in c is connected by an edge in E. A clique can be a single node, an edge, a triangle, etc. The joint probability distribution of an MRF is then given by the product of the potential functions.

$$P(X) = \frac{1}{\mathcal{Z}} \prod_{c \in \mathbb{C}} f_c(x_c), \tag{2.2}$$

where C is the set of cliques of an undirected graph G, X_c for $c \in \mathbb{C}$ is the subset of variables corresponding to the vertices in c, $f_{c \in \mathbb{C}}$ are nonnegative potential functions whose scope is

restricted to the random variables in c, and $\mathcal{Z}$ is the normalizing constant, also known as the partition function, that makes P a probability distribution. For example, of the domain of each random variable is a discrete set, the partition function can be calculated as

$$\mathcal{Z} = \sum_{X} \prod_{c \in \mathbb{C}} f_c(x_c). \tag{2.3}$$

If the potential functions of an MRF are defined only over edges and nodes, i.e., the only cliques with non-constant potential functions are nodes and edges, then we call it a pairwise MRF. And if the domain of all variables in a MRF are continuous, we call it a continuous MRF. A pairwise continuous MRF can be written in the form of:

$$P(X) = \frac{1}{\mathcal{Z}} \prod_{i \in V} \phi_i(x_i) \prod_{(i,j) \in E} \psi_{ij}(x_i, x_j), \tag{2.4}$$

where $\phi_i : \mathbb{R} \to \mathbb{R}_{\geq 0}$ are node potential functions and $\psi_{ij} : \mathbb{R}^2 \to \mathbb{R}_{\geq 0}$ are edge potential functions. $\mathcal{Z}$ is the normalizing constant and can be calculated as $\mathcal{Z} = \int P(X) dX$. Throughout this work, we will assume that the product of potential functions in a MRF is integrable. An example of continuous pairwise MRF is shown in Figure 2.3. For the remainder of this dissertation, we only primarily work with pairwise continuous MRFs unless otherwise specified.

2.2 Inference Tasks in PGMs

Given a PGM, statistical inference seeks to answer probabilistic queries. These inference problems arise naturally in a variety of application domains such as optical flow estimation (Fleet and Weiss, 2006b), image denoising (Lienart et al., 2015), spam detection (Rayana and Akoglu, 2015), etc.

Typical inference tasks involve some kind of high dimensional summation, multiplication, or maximization over sets of random variables. As the number of possible configurations of the random variables grows exponentially in the model size, even computing the partition

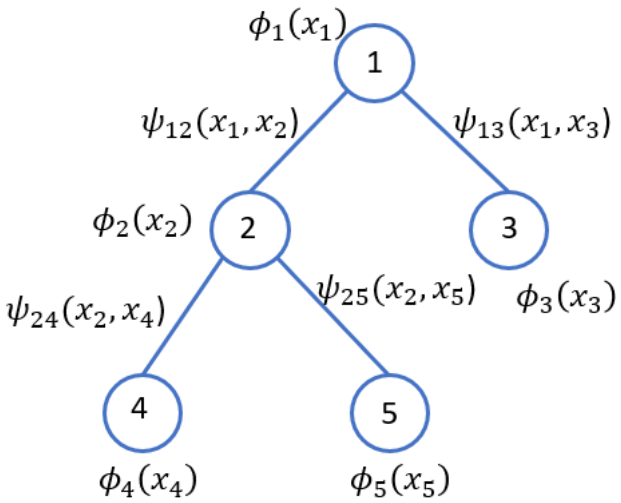


Figure 2.3: Example of an MRF

function, which requires summing over all possible states of the variables is intractable in general (Roth, 1996). To make these models practical, efficient exact/approximate algorithms are needed.

Here, we will consider three main inference tasks in PGMs: marginal inference, maximum a posteriori (MAP) inference, and marginal maximum a posteriori (marginal MAP) inference. Define a partition of random variables to be $X = \{X_S, X_T\}$. The marginal inference problem is the task of computing the marginal probability of a subset of variables by summing out all the other variables, e.g., for a subset of variables X_S, marginal inference is interested in calculating the probability $P(X_S)$

$$P(X_S) = \sum_{X_T} P(X_S, X_T) \tag{2.5}$$

Marginal inference in PGMs is NP-hard (Cooper, 1990) and in fact #P-complete(Roth, 1996), which has inspired the development of approximate inference algorithms for practical applications (Murphy et al., 2013; Yedidia et al., 2005a).

The maximum a posteriori (MAP) estimation problem is to find an assignment to a set of variables that has the highest probability under the model.

$$X^* = \arg\max_X P(X) \tag{2.6}$$

The MAP inference problem is also NP-hard as many natural NP-hard problems can be reduced to MAP inference in an MRF, e.g., see (Park, 2002). Many approximate MAP inference methods have been proposed based on linear programming relaxations, and can perform well in practice even on loopy graphs (Kolmogorov and Wainwright, 2012; Globerson and Jaakkola, 2008; Werner, 2007).

The aim of marginal maximum a posteriori (marginal MAP) estimation is to find the assignment with highest probability of a subset of variables while summing out the remaining variables.

$$X_S^* = \arg\max_{X_S} \sum_{X_T} P(X_S, X_T) \tag{2.7}$$

Marginal MAP is NP-hard even in tree-structured models (Koller and Friedman, 2009). Recent research on marginal MAP problems has focused on the discrete case, e.g., using AND/OR search and NP oracles (Marinescu et al., 2014, 2017; Xue et al., 2016).

2.3 Exact Inference in PGMs

Exact inference in PGMs is intractable in the worst case. The situation is even worse when the variables are continuous: exact inference requires calculating the high dimensional integrals of arbitrary continuous functions, which may not result in explicit closed forms. Despite this, the structure of PGMs can sometimes be exploited to efficiently perform exact inference.

To motivate the role that structure plays in inference, consider the simple discrete MRF model in Figure 2.4. The joint probability is given by

$$p(x_1, x_2, x_3) = \phi_1(x_1)\phi_2(x_2)\phi_3(x_3)\psi_{12}(x_1, x_2)\psi_{23}(x_2, x_3), \tag{2.8}$$

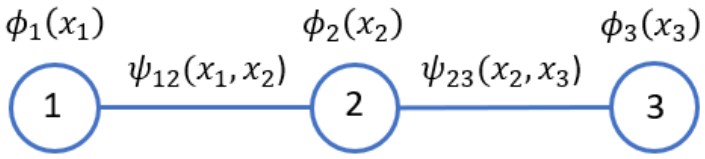


Figure 2.4: Example of an discrete MRF

where each variable can take one of k discrete values. The marginal distribution of $p(x_1)$ can be computed by a straightforward summation.

$$p(x_1) = \sum_{x_2, x_3} \phi_1(x_1)\phi_2(x_2)\phi_3(x_3)\psi_{12}(x_1, x_2)\psi_{23}(x_2, x_3) \tag{2.9}$$

Naïve computation of this summation requires $O(k^2)$ operations, exhausting all possible combinations of x_2 and x_3 values. However, if we change the summation order

$$p(x_1) = \phi_1(x_1) \sum_{x_2} \Big[\phi_2(x_2)\psi_{12}(x_1, x_2) \sum_{x_3} \big[\phi_3(x_3)\psi_{23}(x_2, x_3)\big]\Big], \tag{2.10}$$

then only $O(2k)$ calculations are required to compute the marginal.

The order in which the summations are performed can greatly affect the computational cost. With this motivation, we can first find a good summation (or elimination) ordering, and then perform the summations in that order. This is the motivation behind the variable elimination algorithm Zhang and Poole (1994). Unfortunately, finding an optimal elimination ordering is NP-hard in general. However, a variety of heuristics that can perform well in practice are often used, e.g., minimum degree and minimum fill (Darwiche, 2009).

If a good elimination ordering exists, it can greatly reduce the computational cost of exact inference. However, in general, no elimination ordering may reduce the computational cost enough to make this approach viable in practice. In next two chapters, we will introduce efficient approximate inference methods that also exploit the graph structure, but as they are only approximate, remain tractable even for large PGMs.

CHAPTER 3

MESSAGE-PASSING ALGORITHMS

As exact inference is NP-hard in PGMs, researchers have proposed a variety of approximate inference algorithms to tackle these problems in practice. Message-passing methods, which are based on a dynamic programming representation of the variable elimination method, are a popular alternative. In this approach, directed messages that are functions of local information are passed over edges in the (factor) graph until convergence. In this section, we describe the basic message-passing approach from the belief propagation algorithm to more recent variants applicable in the continuous case.

3.1 Belief Propagation

The belief propagation (BP) algorithm, also known as the sum-product algorithm, was first introduced by Pearl (1982) to solve inference problems in tree-structured PGMs. In BP, a set of functions called messages are defined along the edges in the graph, see Figure 3.1. Messages are updated iteratively in the following form:

$$m_{i\to j}^{t}(x_j) \propto \int \phi_i(x_i)\psi_{ij}(x_i, x_j) \prod_{k\in N(i)\setminus j} m_{k\to i}^{t-1}(x_i)dx_i \tag{3.1}$$

where $N(i)$ is the set of neighbor nodes of node i. The new message $m_{i\to j}^{t}(x_j)$ is computed based on potential functions and the old messages $m_{k\to i}^{t-1}(x_i)$. Marginal approximations, sometimes called pseudomarginals or beliefs, over the nodes and edges can be calculated directly from the potential functions and the messages.

$$\begin{aligned} b_i(x_i) &\propto \phi_i(x_i) \prod_{k\in N(i)} m_{k\to i}(x_i) \\ b_{ij}(x_i, x_j) &\propto \phi_i(x_i)\phi_j(x_j)\psi_{ij}(x_i, x_j) \prod_{k\in N(i)\setminus j} m_{k\to i}(x_i) \prod_{k\in N(j)\setminus i} m_{k\to j}(x_j) \end{aligned} \tag{3.2}$$

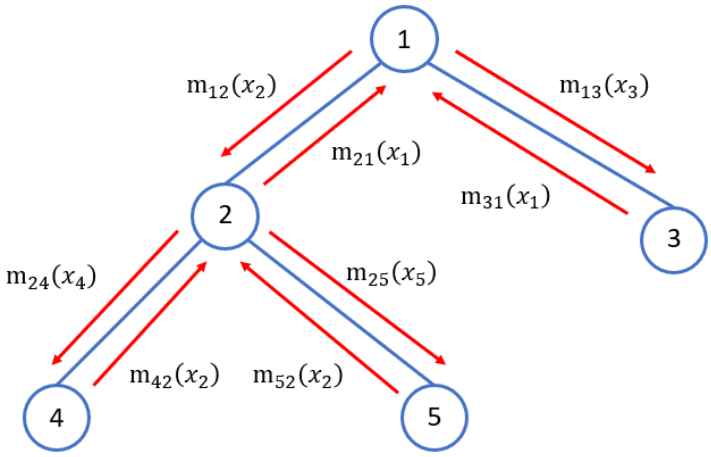


Figure 3.1: Example of message passing schema

The BP method is exact in the case of tree-structured graphical models (Pearl, 1982). That is, at convergence, the above beliefs correspond to the true univariate and pairwise marginals. For general graphs, e.g., graphs with loops), BP is not guaranteed to converge nor does it yield the correct marginals. Despite this, BP can still be applied directly on loopy graphs, where it is called loopy belief propagation (LBP), to yield a reasonable approximation in some cases (Murphy et al., 1999; Taga and Mase, 2006).

Though effective in the discrete case, the BP method cannot, in general, be directly applied in the continuous case without modification (though it can for special cases, e.g., Gaussian graphical models (Weiss and Freeman, 2001)). Consider the message update in (3.1). Calculating the integral with respect to arbitrary an continuous potential function is a nontrivial task. Further, the messages may not be able to be represented in closed form, i.e., the integral for each fixed x_j can be computed but results in an arbitrary univariate continuous function.

3.2 Nonparametric Belief Propagation

A variety of alternatives to BP in the continuous case have been investigated. Sudderth et al. (2003) proposed the Nonparametric Belief Propagation method (NBP). The basic idea

is to approximate the continuous messages as a L-component Gaussian mixtures:

$$m_{ij}(x_j) = \sum_{m=1}^{M} w_j^m \mathcal{N}(x_j; \mu_j^m, \Lambda_j) \tag{3.3}$$

where M is the number of mixtures, w is the vector of mixture weights, which are nonnegative and sum to one, μ and Λ are the parameters of the component Gaussians. Note that, for simplicity, the NBP method only considers Gaussian mixtures with diagonal a covariance matrix.

Message updates in NBP result in product of Gaussian mixtures which is again a Gaussian mixture. However, given a vertex with $d + 1$ neighbors each of which passes a Gaussian mixture message with M mixture components, the method will produce a M^d component Gaussian mixture. As a result, the number of mixture components will grow exponentially with the number of iterations. To handle this, NBP uses an efficient sampling scheme to approximate the M^d component mixture by drawing M independent samples in each message update. This sampling approach is referred to as message-based sampling. Ihler et al. (2005) also proposed a belief-based sampling approach for the NBP method where a single set of samples is drawn from the product of all messages.

Despite these improvements, the message update sampling is still time consuming in practice: the NBP method has complexity of $O(M^D)$ where M is the number of mixture component and D is the maximum degree of the nodes in the graph, which scales poorly when the graph structure has high degree nodes.

3.3 Particle Belief Propagation

A practical improvement over NBP, Particle Belief Propagation (PBP) uses a relatively small number of particles instead of Gaussian mixtures to represent messages and beliefs (Ihler and McAllester, 2009; Frank et al., 2009). The basic idea of the PBP method is to represent the messages via set of points (particles) instead of a continuous function. For each node

in the graph, PBP keeps track of a collection of K sampled particle points. The message updates are evaluated on these particles and the particles are re-sampled, using importance sampling, every iteration. Given a set of sampled particles $\{x_t^{(1)}, x_t^{(2)}, ..., x_t^{(K)}\}$ drawn from proposal function W_t, the message value at particle $x_s^{(i)}$ can be estimated as:

$$m_{t\to s}(x_s^{(i)}) = \frac{1}{K}\sum_{j=1}^{K} \psi_{ts}(x_t^{(j)}, x_s^{(i)}) \frac{\phi_t(x_t^{(j)})}{W_t(x_t^{(j)})} \prod_{u\in N(t)\setminus s} m_{u\to t}(x_t^{(j)}) \tag{3.4}$$

The beliefs can be calculated in similar way as in (3.2) based on the converged messages. Compared to NBP, the key difference is that in PBP, the collection of particles over one node is shared over all of the message products while NBP draws new samples for each message product. The PBP method also does not require the messages to be finitely integrable while NBP method does have this restriction.

PBP has a complexity of $O(|E|K^2)$ per iteration, where $|E|$ is the number of edges in the graph and K is the number of particles per node. But the running time of the PBP method highly depends on the efficiency of the sampling scheme and a proposal function close to the real node marginals can greatly improve the performance of PBP in practice.

3.4 Expectation Particle Belief Propagation

Based on PBP method, Lienart et al. (2015) introduced Expectation Particle Belief Propagation (EPBP). The EPBP method considers the fact that sampling with a proposal from Eq. 3.4 can be expensive. Instead they propose using proposal functions from tractable exponential families. Their proposals are adaptive based on the current node beliefs. EPBP starts tries to approximate the beliefs using tractable distributions in the some simple exponential family, e.g.,

$$q_u(x_u) \propto \eta_{ou}(x_u) \prod_{w\in N(u)} \eta_{wu}(x_u), \tag{3.5}$$

where $\eta_{ou}(x_u)$ and $\eta_{wu}(x_u)$ are elements of the tractable family that approximate $\phi_u(x_u)$ and $\hat{m}_{wu}(x_u)$ respectively. The tractable messages are updating by constructing the cavity distribution $q_u^{\backslash w}$,

$$q_u^{\backslash w}(x_u) \propto q_u(x_u)/\eta_{wu}(x_u) \tag{3.6}$$

and then, using the same strategy as expectation propagation (EP)(Minka, 2001) algorithm, $\eta_{wu}(x_u)$ is updated by minimizing the following KL divergence.

$$\eta_{wu}(x_u) = \underset{\eta \in exp.fam.}{\arg\min}\ KL\big[\hat{m}_{wu}(x_u) q_u^{\backslash w}(x_u) || \eta_u(x_u) q_u^{\backslash w}(x_u)\big] \tag{3.7}$$

The complexity of EPBP is $O(|E|MK)$ per iteration where M is the number of samples used to approximate the message updates. By choosing smaller M, EPBP can be more efficient than PBP.

3.5 Kernel Belief Propagation

Song et al. (2011) proposed a new BP based algorithm to solve the inference problem called kernel belief propagation (KBP). KBP is a joint method that performs learning and inference at the same time. The intuition behind the KBP method is that since calculating closed form for message updates with arbitrary continuous function is intractable in the general case, we can transfer the message update into tractable form where the calculation is obtainable. Given the reproducing kernel Hilbert space (RKHS) (Aronszajn, 1950; Schölkopf et al., 2002), the message update in BP can be expressed as a linear operation in RKHS.

$$m_{ij}(x_j) = \Big\langle \bigotimes_{k \backslash j} m_{ki}, \mathbb{E}_{X_i|x_j}[\xi(X_i)] \Big\rangle_{\mathcal{H}} \tag{3.8}$$

where $\xi(X_i) = \bigotimes_{k\backslash j} \phi(x_i)$ and in RKHS the incoming message product can be written as a single inner product $\bigotimes_{k\backslash j} m_{ki}$. The beliefs then can be calculated as:

$$b(x_i) = P^*(x_i) \prod_{k \in N(i)} m_{ki}(x_i) \tag{3.9}$$

where $P*(x_i)$ is true node marignal distribution estimated using Parzen windows (kernel density estimation).

The cost of one message update is $O(m^2 d_{max})$, where d_{max} is the maximum degree of a node in the graph and m is the number of training data samples. The cost is reasonable for low-degree graphs, but it become very costly for complicated graphs. To compensate, Song et al. (2011) proposed a feature matrix approximation method that uses Gram-Schmidt orthogonalization process (Shawe-Taylor et al., 2004) to reduce the cost of each message update to $O(l^2 d_{max})$ where $l \ll d$.

As KBP does joint inference and learning, it can only be applied in a learning context (not for inference in an existing model). In addition, like all kernel methods, KBP has tunable hyperparameters that can result in sever overfitting if set poorly. Also, the KBP method can return beliefs that are not strictly positive, i.e., they may not be valid distributions. This is, again, a consequence of the kernel mapping.

CHAPTER 4

VARIATIONAL INFERENCE METHODS

A popular alternative to message passing algorithms, variational inference (VI) methods approximate intractable inference tasks by performing them on appropriately chosen surrogate models (Jordan et al., 1999; Blei et al., 2017). The intuition behind VI methods are simple: since the original distribution $p(X)$ is intractable, a more friendly surrogate distribution $q(X)$ is proposed. The optimal surrogate distribution is chosen from a family of (typically tractable) models by minimizing an appropriate notion of distance. Inference problems are then answered by performing inference in the surrogate model where, by design, inference is easier. There are a lot of variational inference methods, including but not limited to mean field approximation (Jaakkola and Jordan, 1998), tree-re-weighted belief propagation (Wainwright et al., 2003), and expectation propagation (Minka, 2013).

4.1 Variational Inference

Given a set of observed variables $x = \{x_1, x_2, ..., x_n\}$, consider computing the the posterior probability distribution $p(y|x)$ of a set of latent variables $y = \{y_1, y_2, ..., y_m\}$ conditioned on x:

$$p(y|x) = \frac{p(y,x)}{p(x)} = \frac{p(y,x)}{\int p(y,x)dz}. \tag{4.1}$$

The integral involved in calculating the $p(x)$, the probability of the evidence, is usually either not available in closed form or intractable. As a result, computing $p(y|x)$ via Bayes rule as above is often computationally infeasible without approximations.

In variational inference, a surrogate distribution from a simpler family $q(y) \in \mathcal{Q}$ is proposed to approximate the conditional distribution $p(y|x)$. Typically, the approximation distribution $q(y)$ will make extra independence assumptions among variables so that $q(y)$

does not have to model the full relationship between x and y. When $\mathcal{Q}$ is chosen to be a more computationally friendly family of distributions, inference on $q(y)$ is much easier than on $p(y|x)$. Notice that a similar method of approximating $p(y|x)$ would be to sample the values of y from $p(y|x)$ with sampling methods, e.g. MCMC sampling method (Chib and Carlin, 1999), and then train a simpler model $q(y)$ on those samples. However, sampling methods in general are very slow and with high variance. In the mean time, variational methods do not require any sampling procedure which are fast and deterministic. The term "variational" is derived from the term "calculus of variations" which is from mathematical analysis meaning choosing the best function in optimization problems.

The aim then is to find an optimal $q(y)$ which can best approximate $p(y|x)$. This is usually accomplished by minimizing the KullbackLeibler divergence (KL-divergence) between two distributions. The KL-divergence measures the difference between two probability distributions (Kullback and Leibler, 1951).

$$KL\Big(P(x)||Q(x)\Big) = -\int P(x)\log\frac{Q(x)}{P(x)}dx \tag{4.2}$$

The KL-divergence is always non-negative and is equal to zero only when the two distributions are identically equal.

Variational inference selects the surrogate model by minimizing the KL-divergence.

$$q^*(y) \in \arg\min_{q\in\mathcal{Q}} KL\Big(q(y)||p(y|x)\Big) \tag{4.3}$$

Consider rewriting the KL-divergence as

$$\begin{aligned} KL\Big(q(y)||p(y|x)\Big) &= -\int q(y)\log\frac{p(y|x)}{q(y)}dy \\ &= \int q(y)\log q(y)dy - \int q(y)\log p(y|x)dy \\ &= \mathbb{E}_{q(y)}\Big[\log q(y)\Big] - \mathbb{E}_{q(y)}\Big[\log p(y|x)\Big] \\ &= \mathbb{E}_{q(y)}\Big[\log q(y)\Big] - \mathbb{E}_{q(y)}\Big[\log\frac{p(y,x)}{p(x)}\Big] \end{aligned}$$

$$= \mathbb{E}_{q(y)}\Big[\log q(y)\Big] - \mathbb{E}_{q(y)}\Big[\log p(y,x)\Big] + \log p(x). \tag{4.4}$$

Note that $\log p(x)$ can be treated as a constant as it does not depend on q. This leads to an optimization problem over what is known as the evidence lower bound (ELBO).

$$ELBO(q) = \mathbb{E}_{q(y)}\Big[\log p(y,x)\Big] - \mathbb{E}_{q(y)}\Big[\log q(y)\Big] \tag{4.5}$$

Minimizing the KL-divergence is equivalent to maximizing the ELBO. As the KL-divergence is always non-negative, we have that $ELBO(q) \leq \log(p(x))$ for all $q \in \mathcal{Q}$ (this is where the ELBO gets its name). We can see that, by construction, maximizing the ELBO yields an estimate of the intractable integral $\log p(x)$.

4.2 Variational Mean Field

The mean field approximation (Anderson and Peterson, 1987) assumes that the surrogate distribution $q(y)$ is a product of independent single variable factors.

$$q(y) = \prod_{j=1}^{m} q_j(y_j) \tag{4.6}$$

Each independent factor $q_j(y_j)$ can take different parametric form, e.g., Gaussians for continuous variables and categorical for discrete variables. Given the simplicity of inference in these types of models, the mean-field approximation is a popular choice for VI methods.

When it comes to optimizing ELBO with mean field approximations, the coordinate ascent variational inference (CAVI) method (Bishop, 2006) is the most commonly used approach. As each variable in the mean field model is independent of the others, we can update the parameter of one factor while keeping the other parameters fixed. In some cases, we can derive the optimal $q_j^*(y_j)$ in closed form by rewriting the ELBO. For example, if all of the latent variables are discrete, the optimal $q_j^*(y_j)$ is proportional to the expectation below, which is obtained by differentiating the ELBO and setting it equal to zero.

$$q_j^*(y_j) \propto \exp\Big(\mathbb{E}_{\prod_{s\neq j} q_s(x_s)}\Big[\log p(y_j|y_{s\neq j}, x)\Big]\Big) \tag{4.7}$$

Since all parameters related to $y_{s\neq j}$ are treated as constants in the right hand side of Eq. 4.7, we can expand to conditional probability to its joint form $p(y, x)$.

$$q_j^*(y_j) \propto \exp\left(\mathbb{E}_{\prod_{s\neq j} q_s(x_s)}\Big[\log p(y_j, y_{s\neq j}, x)\Big]\right) \tag{4.8}$$

Note that the proportionality constant is chosen so that q_j^* sums to one. In this case, the CAVI method will iteratively update each q_j using Eq. 4.8. As each q_j is chosen to maximize the ELBO with the remaining q's fixed, the ELBO can only increase by performing these updates. However, the ELBO is typically a non-concave function of the parameters describing the surrogate models, so the coordinate ascent can get stuck at local optima.

The mean field approximation can be extended by introducing dependencies between the variables, see, for example, structured variational inference (Barber and Wiegerinck, 1999). Bishop et al. (1998) expanded the mean field method to mixtures by adding latent variables within the family.

4.3 Nonparametric Variational Inference

The mean-field approach can yield a poor approximation when the target distribution is multimodal or otherwise has more complicated correlations. As an alternative, Gershman et al. (2012) suggested proposed a new family of variational approximations called nonparametric variational inference (NPVI). Inspired by the nonparametric kernel density estimation method, NPVI chooses the surrogate model to be a mixture of Gaussians over all the variables (note this assumes that the latent variables are continuous).

$$q(z) = \frac{1}{N}\sum_{n=1}^{N} \mathcal{N}(y; \mu_n, \sigma_n^2 I), \tag{4.9}$$

where N is the number of mixtures and $\mathcal{N}(X; \mu_n, \sigma_n^2 I)$ is a multivariate Gaussian distribution with mean μ_n and diagonal covariance matrix $\sigma_n I$, i.e., independent univariate Gaussians.

With this surrogate model, NPVI rewrites the ELBO (Eq. 4.5) into two pieces: an entropy term $\mathcal{H}\Big(q(y)\Big)$ and an expectation term $\mathbb{E}_{q(y)}\Big[p(y,x)\Big]$.

$$\begin{aligned} ELBO(q) &= \mathbb{E}_{q(y)}\Big[\log p(y,x)\Big] - \mathbb{E}_{q(y)}\Big[\log q(y)\Big] \\ &= \mathcal{H}\Big(q(y)\Big) + \mathbb{E}_{q(y)}\Big[\log p(y,x)\Big] \end{aligned} \tag{4.10}$$

Approximations are made separately on each of these two parts. The entropy term is approximated using Jensen's inequality (Huber et al., 2008):

$$\begin{aligned} \mathcal{H}\Big(q(y)\Big) &= -\int q(y)\log q(y)dy \\ &= -\int q(y)\log\frac{1}{N}\sum_{n=1}^{N}\mathcal{N}(y;\mu_n,\sigma_n^2 I)dy \\ &\geq -\frac{1}{N}\sum_{n=1}^{N}\log\int q(y)\mathcal{N}(y;\mu_n,\sigma_n^2 I)dy \end{aligned} \tag{4.11}$$

Note that each of the integrals here compute sums of convolutions of two Gaussian distributions – the convolution of two Gaussians is also a Gaussian.

$$\mathcal{H}\Big(q(y)\Big) \geq -\frac{1}{N}\sum_{n=1}^{N}\log\Big[\frac{1}{N}\sum_{j=1}^{N}\mathcal{N}(\mu_n;\mu_j,(\sigma_n^2+\sigma_j^2)I)\Big] \tag{4.12}$$

Gershman et al. (2012) then approximate the expectation using the multivariate delta method for moments (Bickel and Doksum, 2015). In this approach, $\log p(z,x)$ is approximated by its second order Taylor series expansion at the point μ_n. The approximation is then

$$\mathbb{E}_{q(y)}\Big[\log p(y,x)\Big] \approx \frac{1}{N}\sum_{n=1}^{N}\log p(\mu_n,x) + \frac{\sigma_n^2}{2}Tr(H_n), \tag{4.13}$$

where $H_n = \nabla_z^2 \log p(z,x)$ is Hessian matrix of second derivatives. Finally the approximate ELBO is as below.

$$ELBO(q) \approx \frac{1}{N}\sum_{n=1}^{N}\Big[\log p(\mu_n,x) + \frac{\sigma_n^2}{2}Tr(H_n) + \frac{1}{N}\sum_{j=1}^{N}\mathcal{N}(\mu_n;\mu_j,(\sigma_n^2+\sigma_j^2)I)\Big] \tag{4.14}$$

Optimizing Eq. 4.14 can be done in different ways. Gradient based methods solve this optimization problem but requires calculating the gradients of Eq. 4.13, which can be computationally expensive as it involves computing the third order derivatives. As an alternative, Gershman et al. (2012) proposed to optimize the parameters iteratively using the second order approximation (4.14) for the variances and using a first-order approximation of the ELBO to optimize the means. The method is reported to converge in only a few iterations.

The NPVI method is similar to the approximate VI method we propose here. They differ primarily on the choice of entropy approximation. We will discuss this in more detail in later chapters.

CHAPTER 5

BETHE VARIATIONAL INFERENCE

In this chapter, we will present Bethe Variational Inference, a novel variational method for inference. We begin by introducing the basic idea of our method.

5.1 Bethe Free Energy

It has been shown that the fixed point messages in BP method correspond to the local optima of an optimization problem under constraints (Yedidia et al., 2005b), known as the Bethe Free Energy (BFE). In an MRF $G = (V, E)$ with nonnegative potential functions $\phi_{i\in V}, \psi_{(i,j)\in E}$, we can define a set of probability distributions, also called beliefs, satisfying the following (local) marginalization constraints.

$$\begin{aligned} \forall i \in V, \int b_i(x_i) = 1 \\ \forall (i,j) \in E, \int b_{ij}(x_i, x_j) dx_i = b(x_j) \end{aligned} \tag{5.1}$$

Under the above constraints, the (negative) Bethe Free Energy $F(b)$ is defined as:

$$\begin{aligned} F(b) = &\sum_{i\in V} \mathbb{E}_{b_i(x_i)}[\log \phi_i(x_i) - \log b_i(x_i)] \\ &+ \sum_{(i,j)\in E} \mathbb{E}_{b_{ij}(x_i,x_j)}[\log \psi_{ij}(x_i, x_j) - \log \frac{b_{ij}(x_i, x_j)}{b_i(x_i) b_j(x_j)}] \end{aligned} \tag{5.2}$$

As to the BP method, beliefs in BFE can also be treated as pseudo marginals and used to solve inference problems. Maximizing (5.2) over the beliefs will result in an approximation to the log-partition function $\log \mathcal{Z}$,

$$\log \mathcal{Z} \approx \max_{b\in\mathcal{L}(G)} F(b) \tag{5.3}$$

where $\mathcal{L}(G)$ is the collection of all beliefs satisfying the local marginalization constraints.

Similar to the BP method, when the graph is a tree, the maximum value of (5.2) is equal the true log-partition function and the optimal beliefs are equal to the true node and edge marginals. This does not hold for the BFE on general graphs, where optimizing the BFE only yields an approximation.

While BP based methods are not guaranteed to converge on general graphs, gradient based optimization algorithms using the BFE can yield convergent alternatives, though, in practice, the rate of convergence is often slower than their message-passing counterparts. We note that the (5.2) is not a concave function in general – so gradient methods can become stuck on local optima. There have been efforts made to transform (5.2) into a concave optimizing problem using so-called double counting numbers (Wainwright et al., 2005; Meltzer et al., 2009; London et al., 2015). These approaches introduce a collection of weights that alter the entropy approximation.

5.2 Bethe Variational Inference

Consider a surrogate distribution $b(X)$ in the form of mixture of mean-field distributions as follows.

$$b(X) = \sum_{m=1}^{M} \lambda_m \prod_{i \in V} b_i^m(x_i) \tag{5.4}$$

where M is the number of mixture components, λ_m is the m^{th} mixture weight, and each $b_i^m(x_i)$ is an arbitrary univariate probability distribution. Recall the constraints for BFE in (5.1), these constraints are naturally satisfied by the above surrogate distribution: a mixture of probability distributions is also a probability distribution and marginalization of product of independent univariate distributions is itself a product of independent univariate distributions.

Instead of minimizing the KL-divergence between the surrogate and original distributions, our proposed strategy is to minimize the difference between them by optimizing over the BFE

using beliefs that correspond to marginals of (5.4). That is, we attempt to maximize the BFE by applying gradient ascent over the parameters that describe the beliefs. We call this approximate variational scheme Bethe Variational Inference (BVI).

Compared to the message passing based methods, our method optimizes directly on the beliefs, which obviates the difficulty of calculating the message updates with continuous variables (since the message updates are not always available in closed form). Another advantage of our method is that we can use any integral approximation that we would like to approximate the expectations in the BFE. Our method also does not have requirements on the potential functions, any non-negative normalizable continuous functions will work while many of the message-passing algorithms only work with a more limited set of potential functions. BVI can also handle the mixed PGMs, i.e., models with both continuous and discrete variables, with essentialy the same high-level approach.

Compared to other variational inference methods, the approximation produced by BVI has several advantages. First, in pairwise MRFs, integrals in the BFE are limited to the local marginal polytope so that the order of the integrals are at most two. Standard VI methods optimize a lower bound on the KL-divergence. Depending on the approximation, this can result in high dimensional integrals, which necessitates additional approximations. Second, the BFE yields an exact solution on tree models while other entropy approximations, such the one considered by NPVI based on Jensen's inequality, have no such guarantees.

5.3 Expectation Approximation in BVI

Exact computation of the gradient of the BFE is often not possible in practice if the expectations that result in continuous integrals cannot be calculated in closed form. However, given that these integrals integrals can be treated as expectations with respect to the beliefs, a variety of approximation procedures can be applied. In this section, we will discuss several

possible expectation approximation methods that can be applied as part of the gradient procedure.

5.3.1 Gauss-Hermite Quadrature Method

Numerical quadrature methods provide one source of approximate integration methods. Consider a continuous model in which the the beliefs are chosen to be Gaussian distributions. The node and edges belief are in the form:

$$\begin{aligned} b_i(x_i) &= \sum_{m=1}^{M} \lambda_m \mathcal{N}(X x_i; \mu_i, \sigma_{ii}^2) \\ b_{ij}(x_i, x_j) &= \sum_{m=1}^{M} \lambda_m \mathcal{N}\left(\begin{bmatrix} x_i \\ x_j \end{bmatrix}; \begin{bmatrix} \mu_i \\ \mu_j \end{bmatrix}, \begin{bmatrix} \delta_{ii} & \delta_{ij} \\ \delta_{ij} & \delta_{jj} \end{bmatrix} \right) \end{aligned} \tag{5.5}$$

As any nonnegative, continuous function can be arbitrarily well approximated by a mixture of Gaussians (Scott, 1992), this family is very expressive, but, as a result, more local optima are likely to exist when optimizing the BFE. Since we are integrating against a Gaussian distribution, the Gauss-Hermite Quadrature (GHQ) method is a natural choice (Golub and Welsch, 1969). Gauss-Hermite Quadrature methods are used to approximate integrals in a certain form. Given a real-valued function $f(x)$, GHQ approximates the expectation, $E[f]$, with respect to a Gaussian distribution as:

$$\mathbb{E}_{\mathcal{N}(\mu,\sigma^2)}[f(x)] \approx \sum_{k=1}^{K_Q} \frac{w_k}{\sqrt{\pi}} f\left(\sqrt{2\sigma^2} y_k + \mu\right) \tag{5.6}$$

where K_Q is the number of quadrature points, and the w_k's and y_k's are determined by the GHQ method (and are independent of the mean and variance). For the multivariate case, the integrals can be approximated iteratively, i.e. applying the GHQ method one variable at a time.

GHQ methods come a with strong theoretical guarantee:

Theorem 1 (Golub and Welsch (1969))**.** *For a positive integer K_Q, mean $\mu \in \mathbb{R}$, and variance $\sigma^2 \in \mathbb{R}_{>0}$, GHQ constructs $w_1, \ldots, w_{K_Q} \in \mathbb{R}$ and $y_1, \ldots, y_{K_Q} \in \mathbb{R}$ such that there exists a $\xi \in \mathbb{R}$ with*

$$\mathbb{E}_{\mathcal{N}(\mu,\sigma^2)} f(x) = \sum_{k=1}^{K_Q} \frac{w_k}{\sqrt{\pi}} f\left(\sqrt{2\sigma^2} y_k + \mu\right) + \frac{n!\sqrt{\pi}}{2^n} \frac{f^{(2K_Q)}(\xi)}{(2n)!}$$

As inferred from theorem above, using K_Q quadrature points, the approximation is exact whenever f is a polynomial of degree at most $2K_Q - 1$ in each variable separately.

A known drawback of the BFE is that, in the case of continuous random variables, it need not be bounded from below over the local marginal constraint set. This unboundedness can occur even in Gaussian MRFs (Cseke and Heskes, 2011). This makes BFE potentially undesirable for continuous MRFs in practice. However, for the optimization problem considered here (over a subset of marginals that arise from a joint distribution), we can show that the BFE is bounded from below for Gaussian MRFs.

Theorem 2. *The BFE optimization problem (5.2) is bounded below whenever p is a Gaussian distribution and the optimization is performed over beliefs that arise from any joint distribution q with finite first and second moments (for example, when q is a mixture of Gaussians).*

Proof. Given a Gaussian distribution over n variables $p(x) = \tilde{p}(x)\mathcal{Z}$, $\tilde{p}(x) = \exp(-\frac{1}{2}x^T Jx + h^T x)$, J positive definite, suppose we approximate it by a continuous distribution q, such that $\mathbb{E}_q[X] = \mu, \mathbb{V}_q[X] = \Sigma$ (which are assumed to exist). Denote mutual information by $\mathbf{I}$, entropy by $\mathbf{H}$, and the set of edges in the Gaussian MRF by $\mathcal{E}$. By simple algebra, the BFE is then

$$\begin{aligned}
\mathcal{F}(q) &= \mathbb{E}_q[\frac{1}{2}x^T Jx - h^T x] + \sum_{(i,j)\in\mathcal{E}} \mathbf{I}[q_{ij}] - \sum_i \mathbf{H}[q_i] \\
&\geq \mathbb{E}_q[\frac{1}{2}x^T Jx - h^T x] - \sum_i \mathbf{H}[q_i]
\end{aligned}$$

$$= \frac{1}{2}\text{Tr}[J\Sigma] + \frac{1}{2}\mu^T J\mu - h^T\mu - \sum_i \mathbf{H}[q_i],$$

where the inequality follows from the fact that mutual information is always non-negative. Since J is positive definite, the quadratic form $\frac{1}{2}\mu^T J\mu - h^T\mu$ is bounded from below. So it's sufficient to show that $g(q) \triangleq \frac{1}{2}\text{Tr}[J\Sigma] - \sum_i \mathbf{H}[q_i]$ is bounded from below.

Lemma 3. *Let A, B be two $n \times n$ real symmetric matrices, with B positive definite; let $\lambda_n(A)$ be the smallest eigenvalue of A. Then* $\text{Tr}[AB] \geq \lambda_n(A)\text{Tr}[B]$.

Proof. Denote the eigenvalues of A by $\lambda_1, ..., \lambda_n$, and denote the eigenvalues of B by $\gamma_1, ..., \gamma_n$. Let $A = U\Lambda U^T$, $B = V\Gamma V^T$ be the eigen-decompositions of A and B, such that U, V are orthogonal matrices, and Λ and Γ are diagonal matrices with $\Lambda_{ii} = \lambda_i$ and $\Gamma_{ii} = \gamma_i$ for $i = 1, ..., n$. Then

$$\begin{aligned}\text{Tr}[AB] &= \text{Tr}[U\Lambda U^T V\Gamma V^T] = \text{Tr}[V^T U\Lambda U^T V\Gamma] = \text{Tr}[W^T\Lambda W\Gamma] = \text{Tr}[(\Lambda W)^T W\Gamma] \\ &= \sum_{i,j} \Lambda W \odot W\Gamma = \sum_i \langle \Lambda w_i, \gamma_i w_i\rangle = \sum_i \gamma_i w_i^T \Lambda w_i \geq \sum_i \gamma_i \lambda_{min}(A) \\ &= \lambda_{min}(A)\text{Tr}[B]\end{aligned}$$

where $W := U^T V$ and let w_i be the ith column of W. Note that $\|w_i\| = 1$ since W is orthogonal. The greater or equal sign can be explained by the variational characterization of $\lambda_{min}(A)$, and $\gamma_i > 0$ □

As a result, we have

$$\begin{aligned}g(q) &\geq \frac{\lambda_n(J)}{2}\text{Tr}[\Sigma] - \sum_i \mathbf{H}[q_i] \\ &\geq \frac{\lambda_n(J)}{2}\sum_i \Sigma_{ii} - \frac{1}{2}\sum_i \log(2\pi e\Sigma_{ii}),\end{aligned}$$

where the first inequality follows from Lemma 3 and the second inequality is a consequence of the fact that differential entropy of a distribution with variance σ is maximized by a Gaussian distribution with variance σ. Finally, as $\lambda_n(J), \Sigma_{11}, \ldots, \Sigma_{nn} > 0$, we have that $(\lambda_n(J)\Sigma_{ii} - \log\Sigma_{ii})$ is bounded below for all i. □

5.3.2 Sampling Methods

Sampling provides a generic alternative to quadrature methods that can be efficiently applied whenever the beliefs represent distributions that are easy to sample from. In particular, the beliefs in (5.2) are in the form of a mixture of products of univariate distributions, which are convenient to sample from as long as the univariate distributions are simple to sample from. For the each expectation with respect to beliefs of this form, we can first sample a mixture component based on the mixture weights, then sample from each of the independent distributions separately. Sampling can be inefficient whenever the univariate distributions are not easy to sample from. As a result, many of the standard distributions can be handled in this framework. Take the Dirichlet distribution as an example.

$$Dir(X|\alpha) = \frac{\Gamma(\sum_{i=1}^{K} \alpha_i)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \prod_{i=1}^{K} x_i^{\alpha_i - 1}, \tag{5.7}$$

where $\{x_1, ..., x_K\}$ belongs to the $K-1$ dimensional simplex which means that $x_i \geq 0$ and $\sum_{i-1}^{K} x_i = 1$. The normalization constant involves the Gamma function, $\Gamma(\alpha)$, which is defined as $\Gamma(\alpha_i) = \int x^{\alpha_i - 1} \exp(-x) dx$.

To sample from a Dirichlet distribution, first, draw K independent samples $\{y_1, ..., y_K\}$ from the Gamma distribution:

$$Gamma(\alpha_i, 1) = \frac{y_i^{\alpha_i - 1} exp^{-y_i}}{\Gamma(\alpha_i)} \tag{5.8}$$

Then, normalize $\{y_1, ..., y_K\}$,

$$x_i = \frac{y_i}{\sum_{i=1}^{K} y_i}, \tag{5.9}$$

and $\{x_1, ..., x_K\}$ will follow the Dirichlet distribution $Dir(X|\alpha)$.

Varying the number of samples can greatly affect the performance of sampling approximation. However, the flexibility of our method's choices of independent belief distributions

makes it very easy to choose the most suitable sampling method for expectation approximation. And in practice, we observe that with proper choice of independent belief distributions, only a few sampling points (usually less than 10) could achieve reasonably accurate approximations for our computations. We should note that highly accurate answers may not be necessary here - the sampling procedure, when used to estimate the gradient of the BGE, can be viewed as a stochastic gradient method.

5.4 Gradient Ascent on the BFE

Regardless which approximation scheme is selected, the aim is to apply gradient ascent to optimize the BFE with respect to the parameters of the beliefs. The entire process can be easily vectorized, i.e., coded into matrix operations, making it very efficient to be implemented on modern GPUs. However, there are still a few remaining issues that need to be resolved before applying our method in practice. First, The entropy terms in the BFE for Gaussian mixtures cannot be computed closed form. Thus we will need to approximate it as well. Taylor series expansions have been shown to work well in this case (Huber et al., 2008), so we expect quadrature methods, for example, to perform reasonably well here too. Second, the mixture weights after gradient update need not correspond to a valid probability distribution. To deal with this case, we could apply a projected gradient method. Instead, we utilize a change of variables and represent the mixture weights as a softmax of so that they are non-negative and always sum to one. Finally, when approximating the gradient, we can apply one of two strategies. The first way is to approximate the expectations directly and then calculate the derivatives on the already approximated terms. Another way is to calculate the exact form of derivatives and then approximate on these derivatives. Due to the specific form of BFE definition, the derivatives are also in the form of an expectation. The latter form approximation is more closely related to stochastic gradient methods, and we adopt this approach here.

The complexity of BVI method varies with the form of beliefs as well as the expectation approximation methods. For Gaussian mixture beliefs with the GHQ method, the gradient can be computed in $O(|E|K_Q^2M^2)$ time on a single machine. If stochastic gradient methods are used, the complexity will be reduced to $O(|E|K_Q^2M)$, which can result in better performance with large M. In addition, K_Q can be kept small in practice as long as the log-potential functions can be well-approximated by low-degree polynomials. We will show via experiments that a small number of mixture components is already sufficient in many practical applications.

5.4.1 Derivative calculation of BFE with mixture of independent distributions beliefs

Define the beliefs to be:

$$\begin{aligned} b_i(x_i) &= \sum_m \lambda_m \pi_i^m(x_i) \\ b_{ij}(x_i, x_j) &= \sum_m \lambda_m \pi_i^m(x_i) \pi_j^m(x_j) \end{aligned} \tag{5.10}$$

where $\pi_i^m(x_i)$ is the discrete distribution for variable x_i.

The Bethe Free Energy with the above set of beliefs can be written as

$$\begin{aligned} F(b) = &\sum_{i \in V} \sum_{x_i} b_i(x_i) \log \phi_i(x_i) \\ &+ \sum_{(i,j) \in E} \sum_{x_i} \sum_{x_j} b_{ij}(x_i, x_j) \log \psi_{ij}(x_i, x_j) \\ &- \sum_{i \in V} (1 - N(x_i)) \sum_{x_i} b_i(x_i) \log b_i(x_i) \\ &- \sum_{(i,j) \in E} \sum_{x_i} \sum_{x_j} b_{ij}(x_i, x_j) \log b_{ij}(x_i, x_j) \end{aligned} \tag{5.11}$$

where $N(x_i)$ represents the number of neighbors of node i.

Calculating the partial derivatives of parameter $\pi_i^m(x_i)$ and λ_m.

$$\begin{aligned}\nabla_{\pi_i^m(x_i)} =& \lambda_m \log \phi_i(x_i) \\ &+\lambda_m \sum_{j \in N(x_i)} \sum_{x_j} \pi_j^m(x_j) \log \psi_{ij}(x_i, x_j) \\ &-\lambda_m (1 - N(x_i))(\log b_i(x_i) + 1) \\ &-\lambda_m \sum_{j \in N(x_i)} \sum_{x_j} \pi_j^m(x_j)(\log b_{ij}(x_i, x_j) + 1)\end{aligned} \tag{5.12}$$

$$\begin{aligned}\nabla_{\lambda_m} =& \sum_{i \in V} \sum_{x_i} \pi_i^m(x_i) \log \phi_i(x_i) \\ &+ \sum_{(i,j) \in E} \sum_{x_i} \sum_{x_j} \pi_i^m(x_i) \pi_j^m(x_j) \log \psi_{ij}(x_i, x_j) \\ &- \sum_{i \in V} (1 - N(x_i)) \sum_{x_i} \pi_i^m(x_i)(\log b_i(x_i) + 1) \\ &- \sum_{(i,j) \in E} \sum_{x_i} \sum_{x_j} \pi_i^m(x_i) \pi_j^m(x_j)(\log b_{ij}(x_i, x_j) + 1)\end{aligned} \tag{5.13}$$

We also apply a change of variable technique to $\pi_i^m(x_i)$ and λ_m with the soft-max function so that the updated parameters are still a discrete distribution.

5.5 One Shot Inference

BVI can be viewed as a one shot inference method; meaning that after the gradient update converges, each of the typical inference tasks can be approximated directly from the joint probability distribution generated by the beliefs, perhaps with some small additional computations. This is a significant advantage in practice, especially for those applications in which repeated inference with different query variables is needed.

5.5.1 Marginal Inference

The marginal distribution over a subset variables X_S can be approximated directly from (5.4):

$$p(X_S) = \sum_{m=1}^{M} \lambda_m \prod_{x_i \in X_S} b_i^m(x_i), \tag{5.14}$$

where X_S represents a subset of variables in the model. Since each $b_i^m(x_i)$ is independent of the other beliefs in its mixture component, the marginal distribution $p(X_S)$ under our model assumptions only requires the terms containing variables in X_S. Calculating the values of the marginal distributions is straightforward and efficient.

5.5.2 MAP Inference

MAP inference tries to find the assignment with highest probability (mode of the distribution):

$$X^* = \arg\max_X \sum_{m=1}^{M} \lambda_m \prod_{x_i \in V} b_i^m(x_i) \tag{5.15}$$

Consider the special form of mixture beliefs in our model, this can be done either by using a standard projected gradient ascent method or by applying a gradient ascent method starting from each of the M modes of the mixture components and select the highest probability assignment. An example of MAP inference gradient ascent on an Gaussian mixture model is shown in Fig. 5.1. Notice that starting from the separate modes of each mixture component (blue points in the figure), the gradient ascent method will typically not need many iterations to converge the the mode of the mixture distribution (red point in the figure). Our experimental results also suggest that this method works quite well in practice.

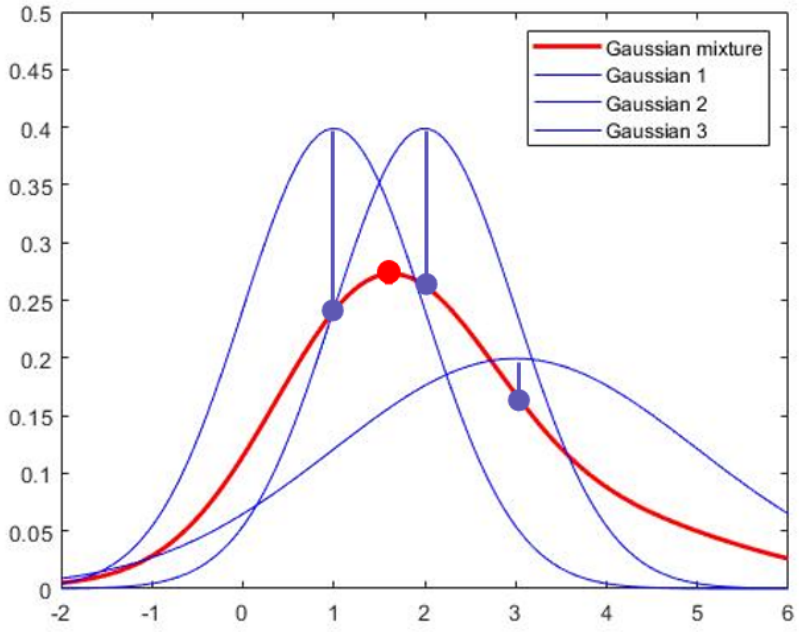


Figure 5.1: Example of MAP inference gradient ascent

5.5.3 Marginal MAP Inference

The marginal MAP inference problem can be regarded as a combination of the previous two inference tasks. For a subset X_S,

$$X_S^* = \arg\max_{X_S} \sum_{m=1}^{M} \lambda_m \prod_{x_i \in X_S} b_i^m(x_i) \tag{5.16}$$

To answer marginal MAP inference queries with our model, we can simply combine the two steps from last two sections. First we get the marignal distribution $p(X_S)$ and then apply gradient ascent on it, starting from each of the mixture modes.

5.5.4 Conditional Marginals

Our model can even achieve desirable conditional marginal distributions with converged beliefs. And inference of conditional marginals for our method is also trivial. Given observed subset of variables X_E, the conditional marginal distribution of subset X_S is given by the

Bayes rules

$$p(X_S|X_E) = \frac{p(X_S, X_E)}{p(X_E)} = \frac{\sum_{m=1}^{M} \lambda_m \prod_{x_i \in X_S \cup X_E} b_i^m(x_i)}{\sum_{m=1}^{M} \lambda_m \prod_{x_i \in X_E} b_i^m(x_i)} \tag{5.17}$$

Since X_E are usually observed, calculating the conditional marginal distributions can be fast. Note that our model with converged beliefs can actually represent any conditional relationships between variables without further processing. This is due to the assumption of our specific belief forms - mixture of independent distributions and this shows another advantage of our proposed method.

5.5.5 Sampling for Generative Models

Even more our method is not only limited to answer inference queries, it can also be used for generative models. Sampling from converged beliefs in our model can be done effectively. If each univariate belief distribution is easy to sample from, then samples can be drawn by first sampling a mixture component according to the mixture weights λ and then sampling from the corresponding univariate independent belief distribution. This is the same procedure for our expectation approximation part.

CHAPTER 6

INFERENCE IN GAUSSIAN MARKOV RANDOM FIELDS WITH MIXTURE OF TREES

In this chapter, we explore the potential possibilities of applying beliefs from mixture of trees model to the BVI algorithm. We first start with introducing the mixture of trees model.

6.1 Mixture of Trees Model

Meila and Jordan (2000) proposed the mixture of trees models which can represent joint probability distributions with a set of mixture weights associated with tree structured PGMs, e.g., Bayesian networks and Markov random fields. A tree structured MRF can always be defined as:

$$T(X) = \frac{\prod_{(i,j)\in E} p_{ij}(x_i, x_j)}{\prod_{i\in V} p_i(x_i)^{deg(x_i)-1}} \tag{6.1}$$

where $p_i(x_i)$ and $p_{ij}(x_i, x_j)$ are unary and pairwise marginal functions. $deg(x_i)$ represents the degree of node x_i in the graph. Following the definition in (6.1), the probability distribution of mixture of trees model can be defined as:

$$P(X) = \sum_{m=1}^{M} \lambda_m T_m(X) \tag{6.2}$$

where M is the number of mixtures and λ_m are nonnegative mixture weights which sum to one $\sum_m \lambda_m = 1$. Tree PGMs $T_m(X)$ are called mixture components and they can have different structures and parameters.

Learning a mixture of trees models can be done by minimizing the KL-divergence using an EM algorithm. However, the learning performance highly depends on the initialization. Kumar and Koller (2009) proposed an efficient algorithm to obtain good initial trees for mixture of trees models by minimizing the α-divergence with $\alpha = \infty$. In the mean time,

inference tasks such as marginal inference and MAP inference can be solved in polynomial time on mixtures of trees. In particular, the marginal distribution of a subset of variables X_S from mixture of trees is a combination of the corresponding marginals from each of the component trees and we take advantages of this feature in our method.

$$P(X_S) = \sum_{m=1}^{M} \lambda_m T_m(X_S) \tag{6.3}$$

However, marginal MAP inference is NP-hard in (mixtures of) tree models and usually approximate algorithms are required in practice (Liu and Ihler, 2013).

6.2 Binary Mixture of Trees Belief

We start from the simple tree case with binary variables to learn the different behavior of optimization algorithms with local polytopes and global marginal polytopes. Each mixture component tree can be defined as

$$T_m(X) = \frac{1}{Z_m} \exp\left(\sum_{(i,j)\in E_m} \theta_{ij}^m(x_i, x_j) \right), \tag{6.4}$$

where the $\theta_{ij}^m(x_i, x_j)$ are the parameters in the model and the marginals can be defined as:

$$\begin{aligned} p_i^m(x_i) &= \sum_{X\setminus x_i} \frac{1}{Z_m} \exp\left(\sum_{(a,b)\in E_m} \theta_{ab}^m(x_a, x_b) \right) \\ p_{ij}^m(x_i, x_j) &= \sum_{X\setminus x_i, x_j} \frac{1}{Z_m} \exp\left(\sum_{(a,b)\in E_m} \theta_{ab}^m(x_a, x_b) \right) \end{aligned} \tag{6.5}$$

The Bethe Free Energy with beliefs corresponding to marginals in a mixture of trees model can be defined as follows.

$$F(b) = \sum_{i\in V} \sum_{x_i} b_i(x_i)[\log \phi_i(x_i) - (1 - N(x_i)) \log b_i(x_i)]$$

$$+\sum_{(i,j)\in E}\sum_{x_i}\sum_{x_j} b_{ij}(x_i,x_j)[\log\psi_{ij}(x_i,x_j)-\log b_{ij}(x_i,x_j)], \tag{6.6}$$

where $N(x_i)$ is the number of neighbors of node i. Define the node and edge beliefs from mixture of trees in the following form.

$$\begin{aligned} b_i(x_i) &= \sum_{m=1}^{M}\lambda_m p_i^m(x_i) \\ b_{ij}(x_i,x_j) &= \sum_{m=1}^{M}\lambda_m p_{ij}^m(x_i,x_j), \end{aligned} \tag{6.7}$$

where λ_m is the mixture weights and $p_i^m(x_i)$ and $p_{ij}^m(x_i,x_j)$ are the corresponding marginals from the mixture component trees.

The derivatives with respect to node and edge beliefs can be calculated as

$$\begin{aligned} \frac{\partial b_i(x_i)}{\partial\theta_{ij}^m(x_i,x_j)} =&\lambda_m\frac{\partial p_i^m(x_i)}{\partial\theta_{ij}^m(x_i,x_j)} \\ =&\lambda_m\frac{-\sum_{X\setminus x_i,x_j}\exp(\sum)}{Z_m^2}\sum_{X\setminus x_i}\exp(\sum)+\lambda_m\frac{1}{Z_m}\sum_{X\setminus x_i,x_j}\exp(\sum) \\ =&-\lambda_m p_{ij}^m(x_i,x_j)p_i^m(x_i)+\lambda_m p_{ij}^m(x_i,x_j) \\ \frac{\partial b_{\setminus i}(\hat{x_i})}{\partial\theta_{ij}^m(x_i,x_j)} =&\lambda_m\frac{\partial p_{\setminus i}^m(\hat{x_i})}{\partial\theta_{ij}^m(x_i,x_j)} \\ =&\lambda_m\frac{-\sum_{X\setminus x_i,x_j}\exp(\sum)}{Z_m^2}\sum_{X\setminus\hat{x_i}}\exp(\sum) \\ =&-\lambda_m p_{ij}^m(x_i,x_j)p_{\setminus i}^m(\hat{x_i}) \\ \frac{\partial b_{ij}(x_i,x_j)}{\partial\theta_{ij}^m(x_i,x_j)} =&\lambda_m\frac{\partial p_{ij}^m(x_i,x_j)}{\partial\theta_{ij}^m(x_i,x_j)} \\ =&-\lambda_m p_{ij}^m(x_i,x_j)p_{ij}^m(x_i,x_j)+\lambda_m p_{ij}^m(x_i,x_j) \\ \frac{\partial b_{\setminus ij}(x_i\hat{,}x_j)}{\partial\theta_{ij}^m(x_i,x_j)} =&\lambda_m\frac{\partial p_{\setminus ij}^m(x_i\hat{,}x_j)}{\partial\theta_{ij}^m(x_i,x_j)} \\ =&-\lambda_m p_{ij}^m(x_i,x_j)p_{\setminus ij}^m(x_i\hat{,}x_j), \end{aligned} \tag{6.8}$$

where $\sum$ are short for the term $\sum_{(i,j)\in E_m}\theta_{ij}^m(x_i,x_j)$.

The derivatives with respect to BFE are then in the following form.

$$
\begin{aligned}
\frac{\partial F(b)}{\theta^m_{ij}(x_i,x_j)} &= \lambda_m p^m_{ij}(x_i,x_j)\big[\log\phi_i(x_i)+\log\phi_j(x_j)\\
&-\sum_{i\in V}\sum_{\hat{x_i}} p^m_i(\hat{x_i})\log\phi(\hat{x_i})\\
&+\log\psi_{ij}(x_i,x_j)-\sum_{(i,j)\in E}\sum_{\hat{x_i}}\sum_{\hat{x_j}} p^m_{ij}(\hat{x_i},\hat{x_j})\log\psi_{ij}(\hat{x_i},\hat{x_j})\big]\\
&-\lambda_m p^m_{ij}(x_i,x_j)\big[(1-N(x_i))(\log b_i(x_i)+1)\\
&+(1-N(x_j))(\log b_j(x_j)+1)\\
&-\sum_{i\in V}\sum_{\hat{x_i}} p^m_i(\hat{x_i})(1-N(\hat{x_i}))(\log b_i(\hat{x_i})+1)\big]\\
&-\lambda_m p^m_{ij}(x_i,x_j)\big[(\log b_{ij}(x_i,x_j)+1)\\
&-\sum_{(i,j)\in E}\sum_{\hat{x_i}}\sum_{\hat{x_j}} p^m_{ij}(\hat{x_i},\hat{x_j})(\log b_{ij}(\hat{x_i},\hat{x_j})+1)\big]\\
=&\lambda_m p^m_{ij}(x_i,x_j)\Big[\log\phi_i(x_i)+\log\phi_j(x_j)+\log\psi_{ij}(x_i,x_j)\\
&-(1-N(x_i))(\log b_i(x_i)+1)-(1-N(x_j))(\log b_j(x_j)+1)\\
&-(\log b_{ij}(x_i,x_j)+1)\\
&+\sum_{i\in V}\mathbb{E}_{p^m_i(\hat{x_i})}\big[(1-N(\hat{x_i}))(\log b_i(\hat{x_i})+1)-\log\phi(\hat{x_i})\big]\\
&+\sum_{(i,j)\in E}\mathbb{E}_{p^m_{ij}(\hat{x_i},\hat{x_j})}\big[(\log b_{ij}(\hat{x_i},\hat{x_j})+1)-\log\psi(\hat{x_i},\hat{x_j})\big]\Big]
\end{aligned}
\tag{6.9}
$$

This effectively expresses the gradient of the BFE in terms of expectations with respect to marginals of the joint mixture of tree structued model. We can transform each component tree into a Bayesian network and sample from it.

6.2.1 Parameter Update with FrankWolfe Algorithm

Updating the parameters of the model, i.e. $p^m_i(x_i)$ and $p^m_{ij}(x_i,x_j)$ can be challenging because the updated distributions almost always do not satisfy the marginalization constraints. Fol-

lowing the discussion in (Belanger et al., 2013), we apply the Frank-Wolfe algorithm to project the updated parameters back to the constraint space. Define μ^{t-1} to be the current set of parameters for all $p_i^M(x_i)$ and $p_{ij}^m(x_i, x_j)$ in one mixture component tree, the Frank-Wolfe algorithm tries to optimize the product of derivative vector $\nabla f(\mu^{t-1})$ and vectors μ over the constraint space M. The intuition is that the optimal parameters $\hat{\mu}$ should be the ones in the constraint space and at the same time closest to the gradient, which means achieving the maximum value under the vector product calculation. The new updated parameters then can be set to be on a point between μ^{t-1} and $\hat{\mu}$.

$$\begin{aligned}\hat{\mu} =& argmax_{\mu \in \mathcal{L}} \langle \nabla f(\mu^{t-1}), \mu \rangle \\ \mu^t =& (1 - \gamma_t)\mu^{t-1} + \gamma_t \hat{\mu},\end{aligned} \tag{6.10}$$

where $\mathcal{L}$ is the set of constraints and γ is the step size, set to $\frac{2}{t+2}$ in our case.

This maximization problem can be further viewed as a MAP inference problem by treating the derivatives of parameters as log node and edge potential functions $\log \phi_i(x_i)$ and $\log \psi_{ij}(x_i, x_j)$.

$$\begin{aligned}\hat{\mu} =& argmax_{\mu \in \mathcal{L}} \langle \nabla f(\mu^{t-1}), \mu \rangle \\ =& argmax_{\mu \in \mathcal{L}} \sum_{i \in V} \sum_{x_i} \mu_i(x_i) \log \phi_i(x_i) \\ &+ \sum_{(i,j) \in E} \sum_{x_i} \sum_{x_j} \mu_{ij}(x_i, x_j) \log \psi_{ij}(x_i, x_j)\end{aligned} \tag{6.11}$$

Any efficient MAP inference algorithm can be applied to solve this problem. Since the graph structure is a tree, we can run the max-product algorithm (Loeliger, 2004) to compute the exact answer in linear time.

6.3 Gaussian Mixture of Trees Belief

For continuous models, we can also make another assumption about the mixture of trees model: We assume that each component tree corresponds to a multivariate exponential

family distribution $P^m(X)$ over a tree-structured graph and only those non-diagonal inverse co-variance elements corresponding to the edges in the tree graph are non-zero. We assume the joint model is a Gaussian.

$$P^m(X) = \frac{1}{Z^m} \exp\big\{ -1/2(X-\mu)^T \Sigma^{-1}(X-\mu)\big\}, \tag{6.12}$$

where $Z^m = (2\pi)^{(-k/2)} det(\Sigma)$ and covariance matrix Σ has to be positive-definite. μ and Σ are known as the moment parameters of Gaussian distributions. And $\Sigma_{ij}^{-1} = 0$ if there are no edges connecting node i and node j.

The marginals of Gaussian distributions can be easily calculated.

$$\begin{aligned} \mu_{ij} &= [\mu_i, \mu_j] \\ \Sigma_{ij} &= [\sigma_{ii}, \sigma_{ij}; \sigma_{ji}, \sigma_{jj}] \end{aligned} \tag{6.13}$$

Note that σ_{ij} is the variance parameter and $\sqrt{\sigma_{ij}}$ represents the standard deviation.

The beliefs then turn out to be as follows

$$b(X) = \sum_m \lambda_m b^m(X) = \sum_m \lambda_m N(X; \mu^m, \Sigma^m) \tag{6.14}$$

The log belief are given by $\log b^m(X) = \log N(X; \mu^m, \Sigma^m)$ and in order to simplify calculation we further define the inverse covariance matrix $\Delta = \Sigma_{ij}^{-1}$.

$$\Delta = \Sigma_{ij}^{-1} = [\delta_{ii}, \delta_{ij}; \delta_{ji}, \delta_{jj}] \tag{6.15}$$

The derivative of $\log b^m(X)$ with respect to the parameters μ_i^m and δ_{ij} can be derived as:

$$\begin{aligned} \frac{\partial \log b^m(X)}{\partial \mu_i^m} &= \sum_j (x_j - \mu_j^m)\Sigma_{ij}^{-1} \\ \frac{\partial \log b^m(X)}{\partial \delta_{ij}^m} &= [\frac{1}{2}\Sigma^m - \frac{1}{2}(X-\mu^m)^T(X-\mu^m)]_{ij} \end{aligned} \tag{6.16}$$

The Bethe Free Energy can be defined as:

$$F(b) = \sum_{i \in V} \int_{x_i} b_i(x_i)[\log \phi_i(x_i) - (1 - N(i)) \log b_i(x_i)]$$

$$+\sum_{(i,j)\in E}\int_{x_i x_j} b_{ij}(x_i,x_j)[\log\psi_{ij}(x_i,x_j)-\log b_{ij}(x_i,x_j)] \tag{6.17}$$

We propose to use a sampling strategy to approximately calculate the derivatives. Following our previous arguments, the derivatives are naturally in the form of expectations with respect to the joint belief $b(X)$ and sampling from our proposed joint beliefs can be implemented efficiently. Given M samples drawn from the joint distribution, $\{X^1, X^2, ..., X^M\} \sim b(X)$, the derivatives can be calculated as follows.

$$\begin{aligned}
\frac{dF(b)}{d\mu_s} =& \frac{d}{d\mu_s}\sum_{i\in V}\int_{x_i} b_i(x_i)\big[\log\phi_i(x_i)-(1-N(i))\log b_i(x_i)\big] \\
&+\frac{d}{d\mu_s}\sum_{(i,j)\in E}\int_{x_i,x_j} b_{ij}(x_i,x_j)\big[\log\psi_{ij}(x_i,x_j)-\log b_{ij}(x_i,x_j)\big] \\
=&\sum_{i\in V}\int_{x_i}\frac{d}{d\mu_s}b_i(x_i)\big[\log\phi_i(x_i)-(1-N(i))(\log b_i(x_i)+1)\big] \\
&+\sum_{(i,j)\in E}\int_{x_i,x_j}\frac{d}{d\mu_s}b_{ij}(x_i,x_j)\big[\log\psi_{ij}(x_i,x_j)-\log b_{ij}(x_i,x_j)\big] \\
=&\sum_{i\in V}\int_{x_i}\int_{X-x_i}\frac{db_i(x_i)}{d\mu_s}\big[\log\phi_i(x_i)-(1-N(i))(\log b_i(x_i)+1)\big] \\
&+\sum_{(i,j)\in E}\int_{x_i,x_j}\frac{db_{ij}(x_i,x_j)}{d\mu_s}\big[\log\psi_{ij}(x_i,x_j)-(\log b_{ij}(x_i,x_j)+1)\big] \\
=&\sum_{i\in V}\int_{x_i}\int_{X\setminus x_i}\frac{db(X)}{d\mu_s}\big[\log\phi_i(x_i)-(1-N(i))(\log b_i(x_i)+1)\big] \\
&+\sum_{(i,j)\in E}\int_{x_i,x_j}\int_{X\setminus x_i,x_j}\frac{db(X)}{d\mu_s}\big[\log\psi_{ij}(x_i,x_j)-(\log b_{ij}(x_i,x_j)+1)\big] \\
=&\sum_{i\in V}\int_{X}\frac{db(X)}{d\mu_s}\big[\log\phi_i(x_i)-(1-N(i))(\log b_i(x_i)+1)\big] \\
&+\sum_{(i,j)\in E}\int_{X}\frac{db(X)}{d\mu_s}\big[\log\psi_{ij}(x_i,x_j)-(\log b_{ij}(x_i,x_j)+1)\big] \\
=&\sum_{i\in V}\int_{X} b(X)\frac{d\log b(X)}{d\mu_s}\big[\log\phi_i(x_i)-(1-N(i))(\log b_i(x_i)+1)\big] \\
&+\sum_{(i,j)\in E}\int_{X} b(X)\frac{d\log b(X)}{d\mu_s}\big[\log\psi_{ij}(x_i,x_j)-(\log b_{ij}(x_i,x_j)+1)\big]
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{i \in V} \mathbb{E}_{b(X)} \Big[\frac{d \log b(X)}{d \mu_s} \big[\log \phi_i(x_i) - (1 - N(i))(\log b_i(x_i) + 1) \big] \Big] \\
&+ \sum_{(i,j) \in E} \mathbb{E}_{b(X)} \Big[\frac{d \log b(X)}{d \mu_s} \big[\log \psi_{ij}(x_i, x_j) - (\log b_{ij}(x_i, x_j) + 1) \big] \Big] \\
&\approx \sum_{i \in V} \sum_{m=1}^{M} \frac{1}{M} \Big[\frac{d \log b(X)}{d \mu_s} \big[\log \phi_i(x_i) - (1 - N(i))(\log b_i(x_i^m) + 1) \big] \Big] \\
&+ \sum_{(i,j) \in E} \sum_{m=1}^{M} \frac{1}{M} \Big[\frac{d \log b(X)}{d \mu_s} \big[\log \psi_{ij}(x_i, x_j) - (\log b_{ij}(x_i^m, x_j^m) + 1) \big] \Big] \qquad (6.18)
\end{aligned}
$$

$$
\begin{aligned}
\frac{dF(b)}{d\delta_{st}} &= \sum_{i \in V} \mathbb{E}_{b(X)} \Big[\frac{d \log b(X)}{d \delta_{st}} \big[\log \phi_i(x_i) - (1 - N(i))(\log b_i(x_i) + 1) \big] \Big] \\
&+ \sum_{(i,j) \in E} \mathbb{E}_{b(X)} \Big[\frac{d \log b(X)}{d \delta_{st}} \big[\log \psi_{ij}(x_i, x_j) - (\log b_{ij}(x_i, x_j) + 1) \big] \Big] \\
&\approx \sum_{i \in V} \sum_{m=1}^{M} \frac{1}{M} \Big[\frac{d \log b(X)}{d \delta_{st}} \big[\log \phi_i(x_i) - (1 - N(i))(\log b_i(x_i^m) + 1) \big] \Big] \\
&+ \sum_{(i,j) \in E} \sum_{m=1}^{M} \frac{1}{M} \Big[\frac{d \log b(X)}{d \delta_{st}} \big[\log \psi_{ij}(x_i, x_j) - (\log b_{ij}(x_i^m, x_j^m) + 1) \big] \Big] \qquad (6.19)
\end{aligned}
$$

In order to maximize the Bethe Free Energy, gradient ascent method with diminishing step size can then be applied until convergence, again using sampling to approximate the partial derivatives.

CHAPTER 7

LEARNING TOPIC MODELS WITH BETHE VARIATIONAL INFERENCE

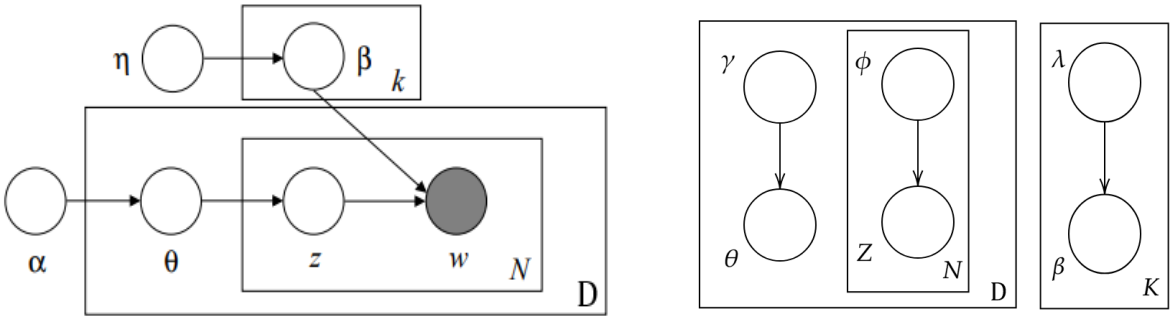
In this chapter, we discuss the potential of applying the Bethe Variational Inference method to topic models. Topic models are statistical models that try to discover latent topics from collection of documents. We start by formulating the learning problem in the lanaguage of probabilistic graphical models.

7.1 Topic Models and Latent Dirichlet Allocation

In Natural Language Processing (NLP), topic models refer to a type of statistical models that discover abstract "topics" from collection of documents (corpus). Latent Dirichlet Allocation (LDA) is an example of a topic model proposed by Blei et al. (2003). LDA assumes that the topic distribution over documents and word distribution over topics are both Dirichlet distributions. In a hierarchical Bayesian model, the joint distribution of the smoothed LDA model can be defined as

$$p(w, z, \theta, \beta|\gamma, \eta) = \prod_{k=1}^{K} p(\beta_k|\eta) \prod_{d=1}^{D} \Big[p(\theta_d|\gamma) \prod_{n=1}^{N_d} p(z_{d,n}|\theta_d) p(w_{d,n}|z_{d,n}, \beta) \Big], \tag{7.1}$$

where K is the number of topics, D is the number of documents, N_d is the number of words in dth document, and V denotes the vocabulary size. We assume $p(\beta_k|\eta)$ and $p(\theta_d|\alpha)$ are Dirichlet distributions: $\beta_k \sim Dir(\eta)$ and $\theta_d \sim Dir(\gamma)$. $\beta_k = (\beta_{k,1}, \beta_{k,2}, ..., \beta_{k,V})$ and $\theta_d = (\theta_{d,1}, \theta_{d,2}, ..., \theta_{d,K})$ are vectors representing distributions of words for topic k and vector representing the distribution of topics for document d respectively. We will denote the collection of all K topics as $\beta = \{\beta_1, ..., \beta_K\}$. $w_{d,n}$ is an integer in {1,...,V} that represents the nth word in dth document while $z_{d,n}$ is an integer in {1,...,K} that indicates the topic of $w_{d,n}$. η and $w_{d,n}$ are treated as given constants. γ is a K dimensional variable that represents the parameters of the corresponding Dirichlet distribution. The graphical representation of



(a) Original LDA model

(b) Surrogate LDA model

Figure 7.1: Bayesian Network Representation of the LDA Model (Blei et al., 2003).

smoothed LDA model is shown in Figure 7.1a. The standard approach to fitting topic models is to perform variational inference using a simpler surrogate model. Blei et al. (2003) proposed a surrogate model by dropping edges that couple θ and β together. We arrive at the graphical model shown in Figure 7.1b so that the joint distribution can be written as:

$$q(\beta_{1:K}, z_{1:V}, \theta_{1,D}|\lambda, \phi, \gamma) = \prod_{i=1}^{K} Dir(\beta_i|\lambda_i) \prod_{d=1}^{D} \left[q(\theta_d|\gamma_d) \prod_{n=1}^{N_d} q(z_{d,n}|\phi_{d,n})\right], \tag{7.2}$$

where λ and γ are Dirichlet parameters and $\phi_{d,n}$ is discrete topic distribution for word $w_{d,n}$. Blei et al. (2003) applied variational inference optimization to minimize the KL-divergence between this surrogate distribution $q(\beta, z, \theta|\lambda, \phi, \gamma)$ and the true posterior $p(\theta, z, \beta|w, \alpha) = \frac{p(w,z,\theta,\beta|\alpha,\eta)}{p(w,\beta|\alpha,\eta)}$ with respect to the variational parameters λ, γ and ϕ.

7.2 Bethe Variational Inference on LDA model

Instead of optimizing on the surrogate model in (7.2), we propose a new surrogate model for the beliefs in the BFE that factorized into distributions on single variables and pairs of variables. The beliefs are treated as pseudo-marginals and optimized over Bethe Free Energy to achieve a surrogate model close to the true marginals.

$$b(\beta, \theta, z) = \sum_{m=1}^{M} \lambda_m \prod_{i=1}^{K} b_{\beta_k}(\beta_k) \prod_{d=1}^{D} \left[b_{\theta_d}(\theta_d) \prod_{n=1}^{N_d} b(z_{d,n}|w_{d,n})\right] \tag{7.3}$$

In this work, we assume the beliefs are mixtures of independent distribution with M mixture components:

$$
\begin{aligned}
b_{\beta_k}(\beta_k) &= \sum_{m=1}^{M} \lambda_m Dir(\beta_k|\alpha_{\beta_k}^m) \\
b_{\beta}(\beta) &= \sum_{m=1}^{M} \lambda_m \prod_{k=1}^{K} Dir(\beta_k|\alpha_{\beta_k}^m) \\
b_{\theta_d}(\theta_d) &= \sum_{m=1}^{M} \lambda_m Dir(\theta_d|\alpha_{\theta_d}^m) \\
b_{w_{d,n}}(z) &= \sum_{m=1}^{M} \lambda_m \pi_{w_{d,n}}^m(z) \\
b_{w_{d,n}}(z,\theta_d) &= \sum_{m=1}^{M} \lambda_m \pi_{w_{d,n}}^m(z) Dir(\theta_d|\alpha_{\theta_d}^m) \\
b_{w_{d,n}}(z,\beta) &= \sum_{m=1}^{M} \lambda_m \pi_{w_{d,n}}^m(z) \prod_{k=1}^{K} Dir(\beta_k|\alpha_{\beta_k}^m)
\end{aligned} \tag{7.4}
$$

where λ_m are the mixture weights, $Dir(\beta_k|\alpha_{\beta_k}^m)$ and $Dir(\theta_d|\alpha_{\theta_d}^m)$ are Dirichlet distributions where $\alpha_{\beta_k}^m$ and $\alpha_{\theta_d}^m$ are the corresponding Dirichlet parameters. $\pi_{w_{d,n}}^m(z)$ is a discrete distribution of topics for each unique word.

The inference procedure in the LDA model can be done by approximating the BFE with respect to proper beliefs. We start from deriving the BFE for the LDA model in (7.1) and assume unique words share the same beliefs:

$$
\begin{aligned}
F(b) = &\sum_{k=1}^{K} \int_{\beta_k} b_{\beta_k}(\beta_k) \log p(\beta_k|\eta) d\beta_k + \sum_{d=1}^{D} \int_{\theta_d} b_{\theta}(\theta_d) \log p(\theta_d|\gamma) d\theta_d \\
&+ \sum_{d=1}^{D} \sum_{n=1}^{N_d} \int_{\theta_d} \sum_{z=1}^{K} b_{w_{d,n}}(z,\theta_d) \log p(z_{d,n}|\theta_d) d\theta_d \\
&+ \sum_{d=1}^{D} \sum_{n=1}^{N_d} \int_{\beta} \sum_{z=1}^{K} b_{w_{d,n}}(z,\beta) \log p(w_{d,n}|z_{d,n},\beta) d\beta \\
&- \sum_{k=1}^{K} \int_{\beta_k} b_{\beta_k}(\beta_k) \log b_{\beta}(\beta_k) d\beta_k
\end{aligned}
$$

$$
\begin{aligned}
&- \sum_{d=1}^{D} \int_{\theta_d} b_{\theta_d}(\theta_d) \log b_{\theta_d}(\theta_d) d\theta_d \\
&- \sum_{d=1}^{D} \sum_{n=1}^{N_d} \sum_{z=1}^{K} b_{w_{d,n}}(z) \log b_{w_{d,n}}(z) \\
&- \sum_{d=1}^{D} \sum_{n=1}^{N_d} \int_{\theta_d} \sum_{z=1}^{K} b_{w_{d,n}}(z, \theta_d) \log \frac{b_{w_{d,n}}(z, \theta_d)}{b_{w_{d,n}}(z) b_{\theta_d}(\theta_d)} d\theta_d \\
&- \sum_{d=1}^{D} \sum_{n=1}^{N_d} \int_{\beta} \sum_{z=1}^{K} b_{w_{d,n}}(z, \beta) \log \frac{b_{w_{d,n}}(z, \beta)}{b_{w_{d,n}}(z) b_{\beta}(\beta)} d\beta
\end{aligned} \tag{7.5}
$$

The BFE can be further viewed as sum of expectations:

$$
\begin{aligned}
F(b) = &\sum_{d=1}^{D} \sum_{n=1}^{N_d} \mathbb{E}_{b_{w_{d,n}}(z)} \Big[\log b_{w_{d,n}}(z) \Big] \\
&+ \sum_{k=1}^{K} \mathbb{E}_{b_{\beta_k}(\beta_k)} \Big[\log p(\beta_k | \eta) - \log b_{\beta_k}(\beta_k) \Big] \\
&+ \sum_{d=1}^{D} \mathbb{E}_{b_{\theta_d}(\theta_d)} \Big[\log p(\theta_d | \gamma) - (1 - N_d) \log b_{\theta_d}(\theta_d) \Big] \\
&+ \sum_{d=1}^{D} \sum_{n=1}^{N_d} \mathbb{E}_{b_{\beta}(\beta)} \Big[\log b_{\beta}(\beta) \Big] \\
&+ \sum_{d=1}^{D} \sum_{n=1}^{N_d} \mathbb{E}_{b_{w_{d,n}}(z,\beta)} \Big[\log p(w_{d,n} | z_{d,n}, \beta) - \log b_{w_{d,n}}(z, \beta) \Big] \\
&+ \sum_{d=1}^{D} \sum_{n=1}^{N_d} \mathbb{E}_{b_{w_{d,n}}(z,\theta_d)} \Big[\log p(z_{d,n} | \theta_d) - \log b_{w_{d,n}}(z, \theta_d) \Big]
\end{aligned} \tag{7.6}
$$

Not surprising, (7.6) is still not a convex function, but we can adopt the stochastic gradient method to converge to a local optimum as is done in typical LDA.

7.3 Expectation Approximation of BFE in LDA model

In order to apply gradient ascent on the BFE, we calculate the derivatives of the parameters. As shown in Appendix A, the derivatives are also in the form of expectations. The exact calculation of expectations in these derivatives is impossible since the vocabulary size can

be tens of thousands of words, resulting in intractable integrals in the expectations. Traditional approximation methods like quadrature approximation and particle methods cannot be applied here because the number of quadrature/particle points required to yield accurate approximations is likely too large considering the dimension of the integrals.

A more efficient way to approximate the expectations is on demand. Sampling is an intuitive choice to approximate the expectations. The beliefs in (7.4) are in the mixture forms which, are convenient to sample from. For the singleton beliefs, we can first sample a mixture component based on the mixture weights, and then sample from either discrete or Dirichlet distributions. Sampling from discrete distribution is trivial while sampling from a Dirichlet distribution is also straightforward (draw independent random samples with each of the parameters from Gamma distribution and normalize them). As for the pairwise beliefs, the sampling routine is similar. After sampling one mixture component, we can sample from the discrete and Dirichlet separately since they are independent of each other.

7.4 MAP Assignment Derivation from Converged Beliefs

In order to calculate the log-likelihood, we need to derive a set of MAP assignment from our converged beliefs, i.e., topics for each word in the corpus $z_{d,n}$, topic distributions for each document θ_d, and word distributions for each topic β_k. We can easily iterate through all possible topics to achieve the MAP assignment for $z_{d,n}$.

$$z^*_{d,n} = \max_z \sum_{m=1}^{M} \lambda_m \pi^m_{w_{d,n}}(z) \tag{7.7}$$

We conduct the gradient ascent again on the log beliefs $\log b(\theta_d)$ and $\log b(\beta_k)$, starting from M mode points corresponding to each mixture component. After convergence, the assignment with highest belief value is chosen as the MAP assignment for θ_d and β_k. We care about the derivative for θ_{di} in the log belief $\log b(\theta_d)$ and the derivative for β_{ki} is similar.

$$\nabla_{\theta_{di}} \log b(\theta_d) = \frac{1}{b(\theta_d)} \sum_{m=1}^{M} \lambda_m \frac{1}{B(\alpha^m_{\theta_d})} \prod_{i=1}^{K} \theta_{di}^{\alpha^m_{\theta_{di}}-1} \cdot \frac{\alpha^m_{\theta_{di}} - 1}{\theta_{di}} \tag{7.8}$$

7.5 Full Bayesian Inference with Bethe Approximation

Our method can also solve the so called full Bayesian inference problems (Vrontos et al., 2000). Most Bayesian inference models focus on optimizing the probability distribution given data, in this section we consider optimizing the pure data probability by maximizing the expectation of the probability distribution. Define the graphical model as follows:

$$P(X) = \prod_{d} \prod_{(i,j)\in E} p(\theta_{ij}^d|\alpha)p(x_i^d, x_j^d|\theta_{ij}^d) \tag{7.9}$$

where D is the given data and E represents the edges in the graph structure which can be learned using Chow-Liu tree algorithm. Based on the joint probability setting, we can define the Bethe Free Energy.

$$\begin{aligned} F(b) =& \sum_{d} \sum_{(i,j)\in E} \sum_{x_i^d, x_j^d} \int_{\theta_{ij}^d} b(x_i^d, x_j^d, \theta_{ij}^d) \log p(x_i^d x_j^d|\theta_{ij}^d) \\ &+ \sum_{d} \sum_{(i,j)\in E} \int_{\theta_{ij}^d} b(\theta_{ij}^d) \log p(\theta_{ij}^d|\alpha_d) \\ &- \sum_{d} \sum_{i\in V} \sum_{x_i} b_i(x_i) \log b_i(x_i) - \sum_{d} \sum_{(i,j)\in E} \sum_{x_i x_j} p(\theta_{ij}^d) \log p(\theta_{ij}^d) \\ &- \sum_{d} \sum_{(i,j)\in E} \sum_{x_i^d, x_j^d} \int_{\theta_{ij}^d} b(x_i^d, x_j^d, \theta_{ij}^d) \log \frac{b(x_i^d x_j^d, \theta_{ij}^d)}{b(x_i^d) b(x_j^d) b(\theta_{ij}^d)} \end{aligned} \tag{7.10}$$

CHAPTER 8

EXPERIMENTAL RESULTS

In the following experiments, we evaluate our method on a variety of inference tasks: marginal inference, MAP inference, and marginal MAP inference problems. We also apply our method on different graph structures, e.g., trees and loopy graphs. The goal is to showcase the performance on different inference tasks, all the potential functions are assumed to be known in advance. The BVI method is implemented using a mixture of Gaussian beliefs, dubbed QBethe. The standard projected gradient ascent with a decaying step size is used in these experiments. Our method is compared with Gaussian expectation propagation (EP) (Minka, 2001), particle belief propagation (PBP) (Ihler and McAllester, 2009), and expectation particle belief propagation (EPBP) (Lienart et al., 2015). All the methods are implemented in MATLAB and run on same machine without parallelization. For PBP, the proposal function is chosen to be the beliefs in the current iteration and Markov chain Monte Carlo (MCMC) sampling method is used for sampling. We sample the initial means and particle points from a normal distribution determined by data.

8.1 Synthetic Tree Experiments

We begin our experiments on a synthetic tree model. The goal of this experiment is to inspect the differences between particle based methods and our method on a simple tree structure with different potential function settings. The tree structure is shown in Figure 8.1, we ran experiments with two different settings of potential functions. In the first experiment, we considered a model in which the node potentials are Gaussian mixtures and the edge potentials are Laplace distributions.

$$\begin{aligned} \phi_u(x_u) &= \alpha_1 \mathcal{N}(x_u - y_u; -1, 1) + \alpha_2 \mathcal{N}(x_u - y_u; 2, 0.5) \\ \psi_{uv}(x_u, x_v) &= \mathcal{L}(x_u - x_v; 0, 1), \end{aligned} \tag{8.1}$$

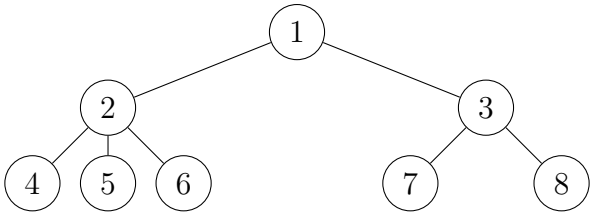


Figure 8.1: Synthetic tree graph.

where $\mathcal{N}(x; \mu, \sigma^2)$ is the standard normal distribution with mean μ and variance σ^2 and $\mathcal{L}(x; \mu, \nu) = \frac{1}{2\nu} \exp(\frac{-|x|}{\nu})$ is the standard Laplace distribution. For our experiments, we set $\alpha_1 = .3$ and $\alpha_2 = .7$.

Loopy Belief Propagation (LBP) method is first run on this model with an evenly spaced 200 point discretization of the interval $[-10, 10]$, to serve as a baseline for all methods. For the particle methods, PBP and EPBP are run with the number of particles, K, set to 10. Generally, increasing the number of particles ensures better results, but the per iteration complexity increases as the number of particles increases. QBethe is run with six mixture components and three quadrature points so that both the particle methods and QBethe have comparable per iteration complexities. We compute the approximate marginals obtained by these algorithms after 20 iterations where one iteration corresponds to updating all messages once. The resulting marginal for each node is displayed in Figure 8.2 (left). In general, the particle methods perform well in this experiment while EP produces poor approximations in all cases. QBethe also appears to underperform slightly compared to the particle methods, and we suspect that this is due to the limited number of quadrature points as well as the difficulty that gradient methods have when a variety of local optima are close together in space and value: the distance between such solutions is less than 10^{-8}, which means that there may be convergence issues near the optimum that may require a large number of iterations to resolve (this is observed empirically).

For our second choice of potentials we select the edge potentials to be multi-modal with two separated peaks, but fix the node potentials to be uni-modal. The goal of this exper-

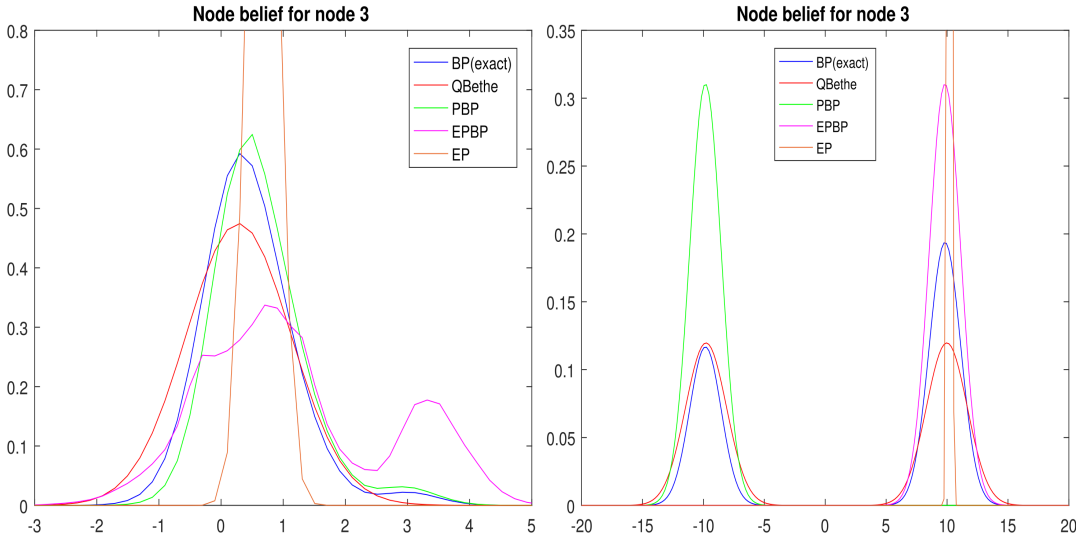


Figure 8.2: Single node marginal beliefs for the eight node tree in which node potentials are multi-modal and edge potentials are uni-modal (left) and single node marginal beliefs for the eight node tree in which node potentials are uni-modal and edge potentials are multi-modal (right).

iment is to illustrate the limitations of the message passing based approaches. Each of the message-passing algorithms will produce approximate univariate marginals that are obtained by multiplying the node potentials by the corresponding messages - this can, and does, result in poor approximations. Specifically, the node and edge potentials are selected as follows.

$$\begin{aligned}\phi_u(x_u) &= \exp(-\frac{|x_u|}{10}) \\ \psi_{uv}(x_u, x_v) &= \exp(\alpha_1(x_u - 10)^2 + \alpha_2(x_v + 10)^2) \\ &\quad + \exp(\alpha_1(x_u + 10)^2 + \alpha_2(x_v - 10)^2)\end{aligned} \tag{8.2}$$

where α_1 and α_2 are both set to be -0.1. The edge potentials are bi-modal functions with peaks at ± 10 and zero mean. We expect that the Gaussian EP algorithm, in particular, will have significant difficulty with these potentials as the mean and mode are very different and the univariate potential functions are not multi-modal.

For this experiment, all algorithm settings are the same as the previous. The marginal distributions obtained by the different algorithms for a fixed node are shown in Figure 8.2

(right). PBP and EPBP only manage to find one of the two peaks in the LBP marginals, though if the number of particles were increased significantly, we expect that particle methods would also yield a multi-modal answer. Gaussian EP places a normal distribution at the mean, resulting in a very inaccurate estimate of the marginal distributions. Contrast this with the first experiment where most of the methods produced a reasonable estimate of the marginals. These observations suggest that, in practice, even for tree structured models, if the shape of the true marginal distributions is unknown, then QBethe may provide a good alternative to the message-passing procedures, at least for comparable per-iteration complexities.

8.2 Marginal Inference with Chow-Liu Trees

The goal of this experiment is to evaluate the performance of different methods in tree-structured, continuous MRFs on the marginal inference task. The reason we restrict ourselves to tree models is that though expensive, the exact ground truth for these tree models can be obtained. While it's very hard to calculate the ground truth for arbitrary continuous potential functions. With the ground truth answers, a more accurate evaluation is possible. Note that tree cases are also the best scenario for the particle based BP methods as they are guaranteed to converge to the optimum on trees. Despite the approximations made by BVI method, we show that QBethe can achieve comparable results with these particle methods on the tree structure models on real world data.

As for the data sets, we select a variety of data sets from the UCI Machine Learning Repository (Dheeru and Karra Taniskidou, 2017) within 4 to 30 variables. The dataset 'B.N.' stands for the Bank Note dataset, 'I.S.E.' stands for Istanbul Stock Exchange, 'Wdbc' is short for Breast Cancer Wisconsin (Diagnostic) Data Set and 'CMSC' is Climate Model Simulation Crashes dataset. We first learned a tree structured model using Chow-Liu tree algorithms (Chow and Liu, 1968) over the continuous variables using Parzen windows (Ni

et al., 2017). However, because of the variances are slightly different, the probabilities produced by this method do not marginalize to each other. We ensure the marginalization conditions are always satisfied for our evaluation purpose. The resulting model is a tree $G = (V, E)$ with a collection of probabilities $p_i(x_i)$ and $p_{ij}(x_i, x_j)$ for each node and edge in the graph. From these probabilities learned with Chow-Liu tree method, We define the potential functions to be:

$$\begin{aligned} \phi_i(x_i) &= p_i(x_i) \prod_{k \in N(i)} M_{ki}(x_i) \\ \psi_{ij}(x_i, x_j) &= p_{ij}(x_i, x_j)/(p_i(x_i)p_j(x_j)M_{ij}(x_j)M_{ji}(x_i)) \end{aligned} \tag{8.3}$$

where each message $M_{ij} : \mathbb{R} \to \mathbb{R}_{>0}$ is an arbitrary continuous function. By this way, the partition function $\mathcal{Z}$ is always one and the real marginal probabilities are exactly the same as the probabilities $p_i(x_i)$ and $p_{ij}(x_i, x_j)$ learned from the Chow-Liu tree method. If the messages are constants and PBP method uses the node beliefs as proposal distributions, then this will result in initializing PBP with the true distributions. To make the problem more realistic, we chose M_{ij} to be:

$$M_{ij}(x_j) = p_j(x_j)^{-1/d_j} \tag{8.4}$$

where d_j is the degree of node j in the graph. In this case, the potential functions become:

$$\begin{aligned} \phi_i(x_i) &= 1 \\ \psi_{ij}(x_i, x_j) &= \frac{p_{ij}(x_i, x_j)}{p_i(x_i)^{1-d_i} p_j(x_j)^{1-d_j}} \end{aligned} \tag{8.5}$$

Other forms of M_{ij} could also work but we found that the performance of all of the methods to be roughly stays the same when varying the reparameterizations.

For these experiments, we run our method (QBethe in figures) from random initialization with $K_Q = 4$ quadrature points and $M = 5$ mixture components. PBP and EPBP are run

with 20 particle points. In this setting, all methods share roughly the same per iteration complexity. The converged beliefs (pseudo marginals) are plugged into BFE to calculate the log-partition function $\log \mathcal{Z}$. Gaussian EP method is not included in these experiments as it produces poor results on all data sets. Note that in all data sets, the exact Z value is one and smaller KL divergence value indicates better approximations. Each method is run for 20 times and the results are reported in the form of $\mu \pm \sigma$ where μ is the mean value and σ represent the standard deviation. The average Z value and the average KL divergence values between the exact and approximate node beliefs over all variables are reported in Table 8.1. For PBP and EPBP, the KL divergence is calculated assuming that the particles are continuous beliefs evaluated at the particle points.

As shown in the table, QBethe significantly outperforms both PBP and EPBP on average on most data sets considering both the partition function estimation and the average univariate KL divergence. This can be explained in the sense that all methods are using roughly the same number of points for integral approximation when calculating the beliefs, QBethe can select optimal points based on quadrature method as suggested in Theorem 1. Another drawback for the particle based methods is that even they do a reasonable job on estimating the marginals, these methods do not guarantee the marginalization of the approximate beliefs. As a result, plugged their marginals into BFE is not accurate. One example is the EPBP results for Wdbc in Table 8.1. Though EPBP has the best performance when estimating the partition function, its KL-divergence is significant worse than QBethe. This suggests that most likely the EPBP method produces beliefs that do not satisfy the marginalization constraints in (5.1). Also increasing the number of particles may seem to help improving accuracy, but comes at significant cost, e.g., PBP requires more than 100 particles ($25\times$ slower) to result in comparable performance comparing to QBethe on the Iris data set.

In an additional set of experiments, we used the same UCI data sets and similarly derived potential functions but changed from Chow-Liu trees, which are often close to star graphs, to

chain-structured models. Note that chain structure is also a tree structure. For data sets with a small number of variables, the chain structure that maximizes the likelihood was chosen while a greedy procedure that iteratively adds the edge with maximum mutual information to the chain is used for larger data sets. The purpose of this investigation was to determine if there is a significant effect on the performance of the methods based on the actual structure of the trees. For example, we might think that errors in the message-passing algorithms may propagate and reduce performance in chain-structured models. For this experiment, the parameters of each method remained the same as the previous experiment, and the results are described in Table 8.2. Again, QBethe outperforms the sampling methods on average, but surprisingly, there does not appear to be a significant difference between the two different structures in terms of quality of approximation. This suggests that the quality of the approximation, at least in these data sets, is largely determined by the actual potential functions rather than the underlying structure. In summary, while all methods can achieve reasonable results on the KL-divergence, if the aim is to compute $\log Z$, QBethe appears to be the better choice in practice.

8.3 MAP Inference with Image Denoising

In this experiment, we consider an image denoising MAP inference problem (Lienart et al., 2015). The edge potentials in this case are not integrable which makes the problem even harder. The aim of this experiment is to demonstrate that BVI method can perform well on medium sized models (thousands of continuous variables) in practical. Our method outperforms the particle methods on this task. The input of this denoising model is a 50×50 image that has been corrupted with Gaussian noise. The corresponding MRF model is a 50×50 grid graph with each pixel treated as a node. The potential functions are chosen to be:

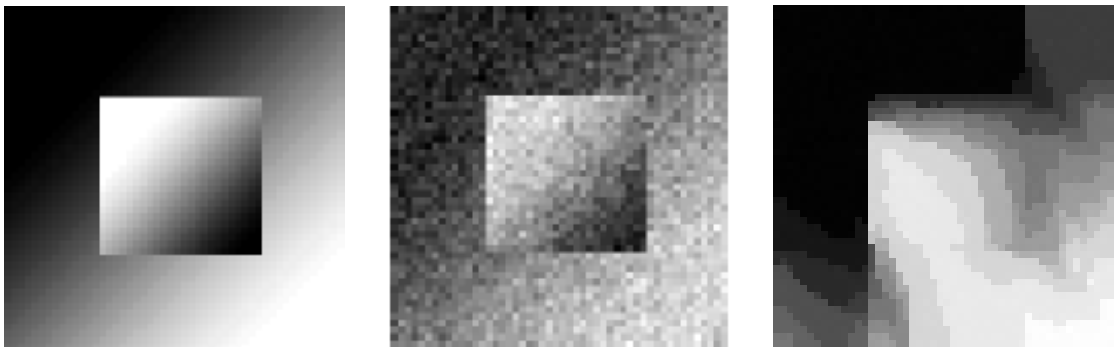
$$\phi_u(x_u) = \mathcal{N}(x_u - y_u; 0, 0.01)$$

Table 8.1: Marginal inference on tree-structure models. All numbers are rounded to two decimal places. In all cases, $Z = 1$.

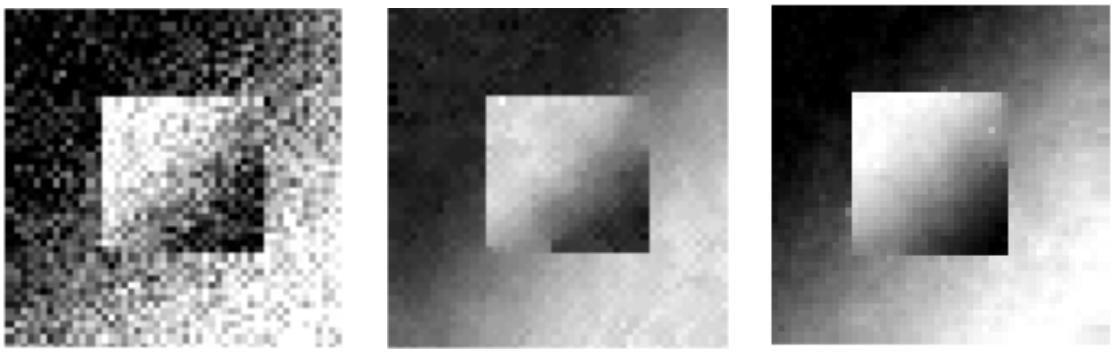
	Average Z			**Average Univariate KL divergence**		
Dataset	**PBP**	**EPBP**	**QBethe**	**PBP**	**EPBP**	**QBethe**
Iris	0.20 ± 0.17	0.37 ± 0.16	$\mathbf{0.97 \pm 0.02}$	0.35 ± 0.33	0.25 ± 0.14	$\mathbf{0.00 \pm 0.00}$
B.N.	0.15 ± 0.18	0.00 ± 0.00	$\mathbf{0.87 \pm 0.01}$	0.62 ± 0.59	0.83 ± 0.01	$\mathbf{0.06 \pm 0.00}$
I.S.E.	0.00 ± 0.01	0.06 ± 0.00	$\mathbf{0.54 \pm 0.02}$	0.78 ± 0.37	0.30 ± 0.00	$\mathbf{0.21 \pm 0.05}$
Seeds	0.12 ± 0.12	0.49 ± 0.15	$\mathbf{0.84 \pm 0.05}$	0.29 ± 0.18	0.12 ± 0.03	$\mathbf{0.02 \pm 0.01}$
Yeast	0.04 ± 0.12	0.00 ± 0.00	$\mathbf{0.67 \pm 0.07}$	3.31 ± 3.61	1.18 ± 0.09	$\mathbf{0.24 \pm 0.05}$
Wdbc	0.05 ± 0.18	0.27 ± 0.20	$\mathbf{0.21 \pm 0.06}$	$\mathbf{0.10 \pm 0.07}$	0.58 ± 0.14	0.18 ± 0.19
Letter	0.00 ± 0.00	0.00 ± 0.00	$\mathbf{0.26 \pm 0.05}$	0.57 ± 0.26	0.73 ± 0.01	$\mathbf{0.07 \pm 0.02}$
Poker	0.62 ± 0.12	0.01 ± 0.00	$\mathbf{0.63 \pm 0.05}$	$\mathbf{0.02 \pm 0.01}$	0.32 ± 0.00	0.06 ± 0.01
CMSC	0.32 ± 0.08	0.47 ± 0.01	$\mathbf{0.56 \pm 0.02}$	0.03 ± 0.01	0.02 ± 0.00	$\mathbf{0.02 \pm 0.00}$

Table 8.2: Marginal inference on tree-structure models. All numbers are rounded to two decimal places. In all cases, $Z = 1$.

	Average Z			**Average KL divergence**		
Dataset	**PBP**	**EPBP**	**QBethe**	**PBP**	**EPBP**	**QBethe**
Iris	0.19 ± 0.11	0.39 ± 0.16	$\mathbf{0.95 \pm 0.02}$	0.22 ± 0.10	0.20 ± 0.12	$\mathbf{0.00 \pm 0.00}$
B.N.	0.75 ± 0.10	$\mathbf{0.92 \pm 0.01}$	0.92 ± 0.03	0.02 ± 0.01	$\mathbf{0.01 \pm 0.00}$	0.01 ± 0.01
I.S.E.	0.00 ± 0.01	0.06 ± 0.00	$\mathbf{0.70 \pm 0.01}$	0.78 ± 0.37	0.30 ± 0.00	$\mathbf{0.08 \pm 0.02}$
Seeds	0.08 ± 0.10	0.16 ± 0.04	$\mathbf{0.92 \pm 0.01}$	0.38 ± 0.21	0.39 ± 0.05	$\mathbf{0.00 \pm 0.00}$
Yeast	0.00 ± 0.00	0.00 ± 0.00	$\mathbf{0.67 \pm 0.02}$	4.71 ± 4.09	1.18 ± 0.14	$\mathbf{0.20 \pm 0.04}$
Wdbc	0.00 ± 0.00	0.00 ± 0.00	$\mathbf{0.19 \pm 0.01}$	0.50 ± 0.25	1.58 ± 0.07	$\mathbf{0.23 \pm 0.02}$
Letter	0.00 ± 0.00	0.00 ± 0.00	$\mathbf{0.20 \pm 0.02}$	0.50 ± 0.50	0.64 ± 0.01	$\mathbf{0.08 \pm 0.02}$
Poker	0.60 ± 0.09	0.06 ± 0.00	$\mathbf{0.60 \pm 0.06}$	$\mathbf{0.02 \pm 0.01}$	0.16 ± 0.00	0.05 ± 0.01
CMSC	0.35 ± 0.09	0.44 ± 0.01	$\mathbf{0.58 \pm 0.03}$	0.03 ± 0.01	0.02 ± 0.00	$\mathbf{0.01 \pm 0.00}$



(a) Original (b) Noisy (c) EP



(d) PBP (e) EPBP (f) QBethe

Figure 8.3: Approximate denoising of the 50×50 image (b).

$$\psi_{uv}(x_u, x_v) = \mathcal{L}^{\lambda}(x_u - x_v; 0, 0.03), \tag{8.6}$$

where y_u is the observed noisy data and $\mathcal{L}^{\lambda}(x; \mu, \nu)$ is a truncated Laplace distribution.

$$\mathcal{L}^{\lambda}(x; \mu, \nu) = \begin{cases} \mathcal{L}(x; \mu, \nu), & |x| \leq \lambda \\ \mathcal{L}(\lambda; \mu, \nu), & otherwise \end{cases}. \tag{8.7}$$

In the experiments, λ was set to 0.2 to be the same as in prior work (Lienart et al., 2015). The number of particles for PBP and EPBP is set to 100. QBethe is run with $M = 1$, i.e., single Gaussian beliefs and three quadrature points. For all the method, the mode (mean value of Gaussian distribution) of the approximate node marginals was selected as the denoised value for each node.

Figure 8.4 shows the denoising results of all methods in grayscale in which the values for all pixels were scaled into the interval $[0, 1]$. Gaussian EP produces a poor estimation

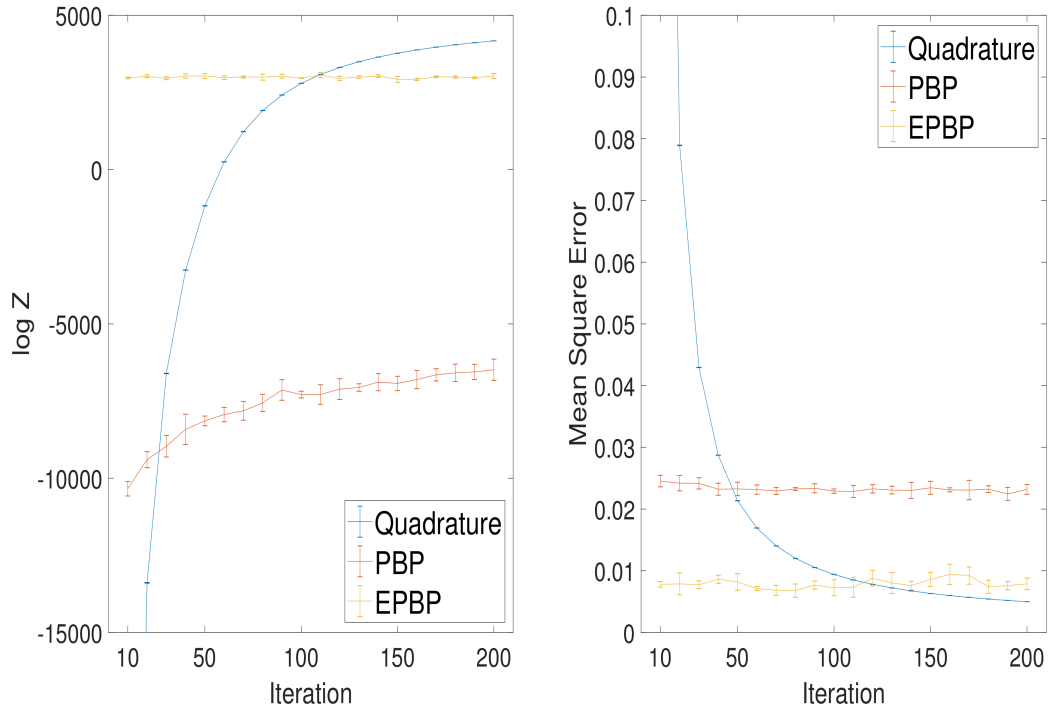


Figure 8.4: Estimate of the partition function and mean square error of EPBP and QBethe for the image denoising problem.

which suggests that the true marginal distributions are mult-modal since the Gaussian EP method only matches the first two moments. QBethe achieves the best performance as the denoising image is more clear and closest to the original one. Figure 8.4 shows the comparison of the particle methods and QBethe method, dubbed Quadrature, with respect to each iteration. The value of the log-partition function and the mean-squared error (MSE) are calculated. Even though QBethe method starts at a significantly worse initialization, it quickly outperforms PBP method and EPBP method both in terms of the log-partition function and the MSE. It seems that QBethe needs more iterations to achieve comparable results to the particle based methods. However, due to the per iteration complexity, QBethe can achieve comparable solutions in roughly the same amount of time. The average per iteration running times are quite different: QBethe (.02s), EPBP (50s) and PBP (305s). As a result, QBethe can run for 2500 iterations in the same amount of time as EPBP run for

Table 8.3: Average log-partition function on a 3-cycle of PBP method. M for number of particle points. The exact log-partition function of this model is -16.17.

PBP	$M = 5$	$M = 25$	$M = 50$	$M = 75$	$M = 100$
$\log Z$	-47.34 ± 38.03	-25.57 ± 31.05	-22.24 ± 19.05	-20.64 ± 19.34	-10.61 ± 12.62

Table 8.4: Average log-partition function on a 3-cycle of QBethe method. M for number of mixture components. The exact log-partition function of this model is -16.17.

QBethe	$M = 2$	$M = 3$	$M = 4$	$M = 5$	$M = 6$
$\log Z$	-17.56 ± 0.36	-17.32 ± 0.42	-17.07 ± 0.45	-16.88 ± 0.41	-16.78 ± 0.42

only one iteration. This indicates that QBethe is more likely to perform better on larger scale problems.

8.4 3 Node Cycle Graph

The motivation for this experiment is that, while the particle based methods can perform reasonable well on a tree, we expect that errors in estimating the individual messages combined with the convergence issues of message-passing algorithms on loopy graphs may combine together to yield very high variance on graphs with cycles. We use a simple 3 node cycle graph to demonstrate this.

For this experiment, the node and edge potentials are carefully chosen to be in certain form so that the true univariate marginal distributions are with three separated modes:

$$\begin{aligned} \phi_u(x_u) &= e^{-0.1|x_u|} \\ \psi_{uv}(x_u, x_v) &= f_{10}(x_u, x_v) + f_{-10}(x_u, x_v) \end{aligned} \tag{8.8}$$

where function f are defined as follows.

$$f_a(x_u, x_v) = e^{-0.1(x_u - a)^2 - 0.1(x_v + a)^2} \tag{8.9}$$

Figure 8.5 shows the true univariate marignal with three separated modes. As also shown in the same figure: PBP brely captures two modes and EPBP only manages to capture two

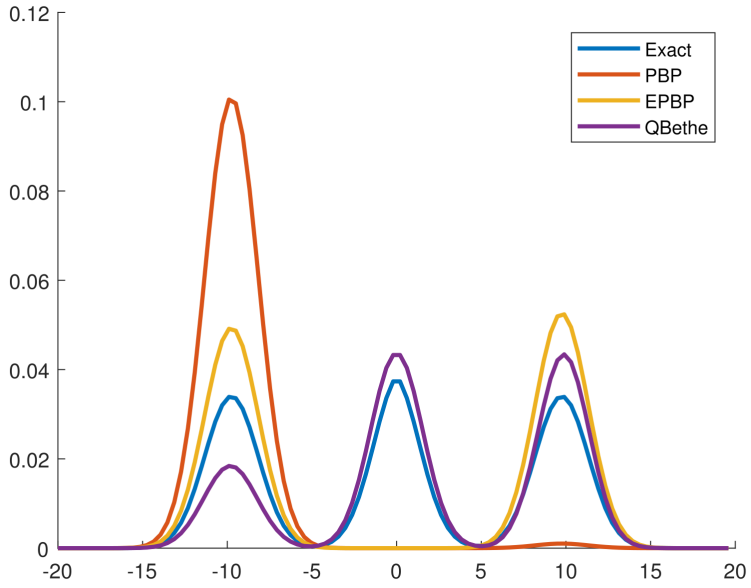


Figure 8.5: One node belief on the three node cycle

modes well. QBethe is the only method that can capture all three modes in this case. We also examined the value of the partition function produced by QBethe with different numbers of mixture components and compared with PBP with different numbers of particle points. We computed the average log-partition function by running 50 trials of each method to 150 iterations. The results of the PBP method are shown in Table 8.3 and the QBethe method in Table 8.4. As the increase of number of particle points, the performance of PBP does improve, but the variance remains large with even 100 particles. A closer inspection on the messages produced by the PBP method suggests that the method is not converging. We also varied the sampling procedure to rejection sampling with a Gaussian proposal to reduce the variance, but this did not seem to improve convergence. QBethe produces a much better approximation on this cycle graph and with more mixture components the method approaches the exact solution. This experiment shows that particle based methods suffer on loopy graphs with certain potentials while the QBethe is more robust to the graph structure as well as different potential functions.

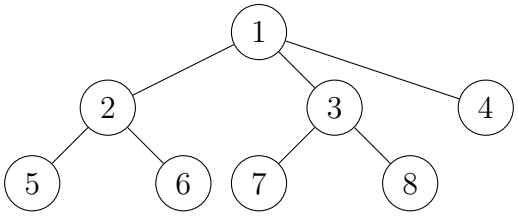
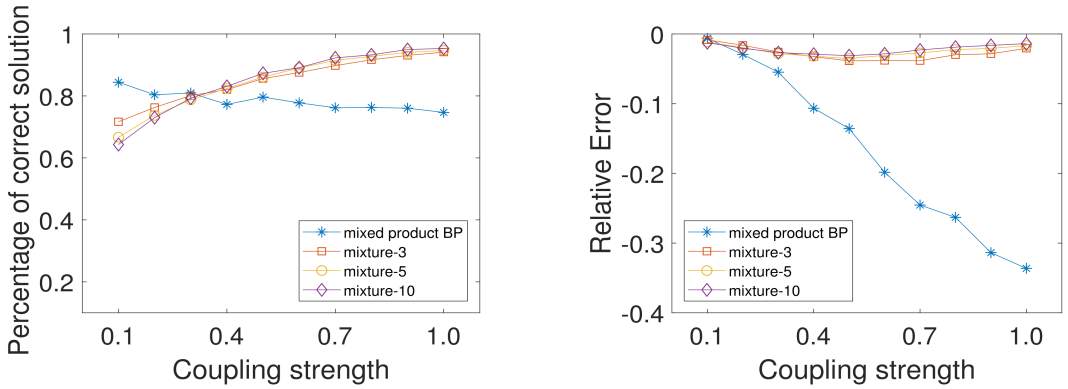


Figure 8.6: Synthetic Tree for Marginal MAP



(a) Percentage of correct solutions (b) Relative error

Figure 8.7: Marginal inference results for synthetic tree model.

8.5 Marginal MAP on Synthetic Trees

This experiment is to demonstrate BVI's performance on marginal MAP inference tasks. We compare our method with a state-of-the-art Marignal MAP algorithm called mixed product BP (MPBP) (Liu and Ihler, 2013). We start from a synthetic tree model with discrete random variables. The synthetic tree structure is shown in Figure 8.6 and probability model is defined as:

$$p(X) = \frac{1}{\mathcal{Z}} \exp\left(\sum_{i \in V} \theta_i(x_i) + \sum_{(i,j) \in E} \theta_{ij}(x_i, x_j)\right) \tag{8.10}$$

The parameters θ_i and θ_{ij} are sampled from Gaussian distributions, $\theta_i(x_i) \sim \mathcal{N}(0, 0.01)$ and $\theta_{ij}(x_i, x_j) \sim \mathcal{N}(0, \sigma^2)$, where σ is varied from 0.1 to 1.0 and called the coupling strength. 100 different set of θ parameters are sampled for each different σ. And we pick three nodes to be MAP nodes and the rest to be marginal (sum) nodes. We consider all $\frac{(8)!}{(3)!(8-3)!} = 56$

MAP/sum nodes combinations and the reported results are averaged over all these combinations. Note that inference using BVI only has to be run once for all combinations while MPBP has to rerun for each of the 56 possibilities. BVI is run with 3, 5, and 10 mixture components, dubbed mixture-3, mixture-5 and mixture-10, for 500 iterations each. We initialize the BVI method with uniform mixture weights and random initial discrete beliefs. We initialize MPBP with random initial messages and run it until convergence[1].

We report the average percentage of correctly identified MMAP assignments in Figure 8.7a and the average relative error in Figure 8.7b. Given the optimal MMAP assignment x_B^*, the relative error $RE(\hat{x}_B, x_B^*)$is defined as:

$$RE(\hat{x}_B, x_B^*) = \frac{p(\hat{x}_B) - p(x_B^*)}{p(x_B^*)}, \tag{8.11}$$

where $\hat{x}_B$ is the estimated MMAP assignment. The Relative Error is non-positive due to the fact that optimal MMAP will always have the highest probability. BVI finishes within 40 seconds while MPBP method requires roughly 250 seconds. This matches our intuition as MPBP method has to be rerun for each MAP/sum node combination. BVI performs better with respect to both the percentage of correct solutions and the relative error on harder problems (larger coupling strength). Further looking into the results shows us that for smaller coupling strength, though the BVI method achieves a worse percentage of correct solutions, the method is missing very close, i.e., the incorrect solution also has very high probability though not optimal. This is likely a consequence the construction of the BVI method: it is designed to return a good approximate distribution not a specific MAP assignment. As the coupling strength does to zero, many solutions are near optimal. Note that, in practice, a solution close to optimal may be sufficient, depending on the application.

[1]At least 50 iterations, if not converged, we run another 200 extra iterations

Table 8.5: Relative error of OSI and mixed-product BP for a marginal MAP task on various UCI datasets.

Dataset	**mixed-BP**	**mixture-3**	**mixture-5**	**mixture-10**
Iris	-0.1848 ± 0.0743	-0.1242 ± 0.1238	-0.1473 ± 0.1326	$\mathbf{-0.1188 \pm 0.0566}$
Letter	-0.0236 ± 0.0380	-0.0253 ± 0.0312	-0.0223 ± 0.0290	$\mathbf{-0.0216 \pm 0.0273}$
S.F.	-0.0404 ± 0.0620	-0.0370 ± 0.0654	-0.0369 ± 0.0664	$\mathbf{-0.0367 \pm 0.0627}$
M.M.	-0.1736 ± 0.2014	-0.1884 ± 0.1892	$\mathbf{-0.1637 \pm 0.1614}$	-0.1732 ± 0.1784
T.T.T.	-0.0789 ± 0.0768	-0.1078 ± 0.0750	-0.0791 ± 0.0562	$\mathbf{-0.0757 \pm 0.0778}$
Y.H.	-0.0590 ± 0.1371	-0.0245 ± 0.0538	-0.0245 ± 0.0538	$\mathbf{-0.0211 \pm 0.0514}$

8.6 Marginal MAP ON UCI Datasets

We also evaluate our method with different number of mixtures on marginal MAP problems over several UCI repository datasets: Iris, Letter, Solar Flare (S.F.), Mammographic masses (M.M.), Tic-Tac-Toe (T.T.T.) and Yacht Hydrodynamics (Y.H.). For each of the data set, we learn a separate discrete tree-structured distribution with the Chow-Liu Tree method (Chow and Liu, 1968). The number of random variables for these data sets ranges from 5 to 17.

We compared our method with Mixed Product BP method. We use 3, 5, and 10 mixture components, dubbed mixture-3, mixture-5 and mixture-10. The performance is examined by calculating the relative MAP assignment error against ground truth. We choose 3 nodes as MAP nodes and treat the rest as marginal/sum nodes, $\frac{(N)!}{(3)!(N-3)!}$ combinations for data set with N variables. The results are averaged over all combinations in the form of $\mu \pm \sigma$ where μ is the mean value and σ the standard deviation. Each method is run for 5 trials with random initialization. The results are shown in Table 8.5.

8.7 Marignal MAP on UAI Challenge Datasets

In this final set of experiments of this chapter, we evaluate our method on several marginal MAP UAI challenge data sets. For each of these discrete models, we considered three different configurations of MAP and sum nodes as in Figure 8.8, 8.9 and 8.10. The graph is a 20×20

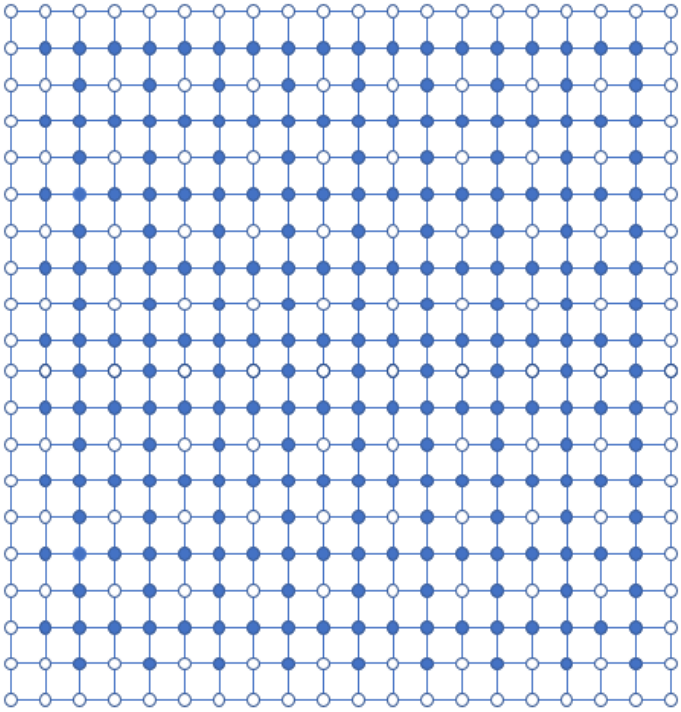


Figure 8.8: MAP/sum nodes combination 1 for UAI challenge datasets. Shaded for sum nodes and unshaded for MAP nodes

grid graph. We compare our method with 5 mixture components, dubbed mixture-5, and Mixed Product BP method. We use variable elimination to conduct exact inference on the model as ground truth. The results are described in Table 8.6. Our method significantly outperforms MPBP in this case. This again shows our method's better robustness against loopy graphs. One reason that message passing based methods suffer on graphs with cycles is that iterative updates of messages will propagate the error in the existing messages which results in never convergence. While our variational method does not have this disadvantages and makes it a better choice for continuous inference tasks on general graphs (graphs with loops).

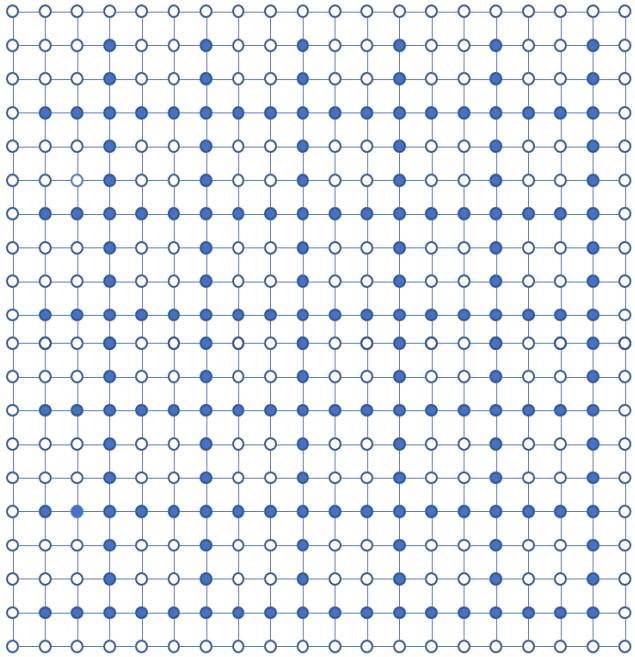


Figure 8.9: MAP/sum nodes combination 2 for UAI challenge datasets. Shaded for sum nodes and unshaded for MAP nodes

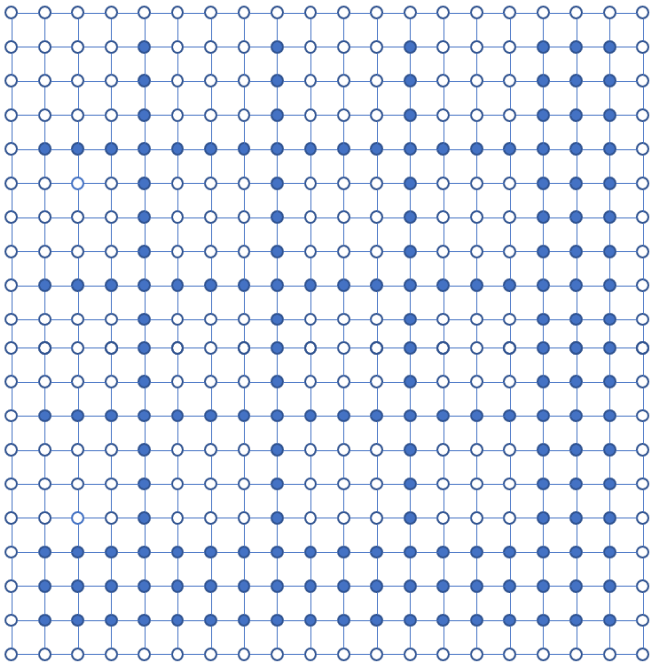


Figure 8.10: MAP/sum nodes combination 3 for UAI challenge datasets. Shaded for sum nodes and unshaded for MAP nodes

Table 8.6: The marginal MAP value produced by mixed-product and OSI on several UAI challenge problems.

	Combination #1		**Combination #2**		**Combination #3**	
Dataset	**mixed-BP**	**mixture-5**	**mixed-BP**	**mixture-5**	**mixed-BP**	**mixture-5**
Grids26	4378.0 ± 149.9	$\mathbf{4963.0 \pm 40.6}$	3927.0 ± 310.3	$\mathbf{4621.6 \pm 60.9}$	3530.5 ± 276.1	$\mathbf{4189.1 \pm 63.8}$
Grids28	6661.1 ± 244.4	$\mathbf{7442.5 \pm 29.2}$	5484.1 ± 412.6	$\mathbf{6944.6 \pm 56.3}$	4851.9 ± 271.6	$\mathbf{6285.6 \pm 16.6}$
Grids29	2280.8 ± 85.4	$\mathbf{2472.2 \pm 24.4}$	2000.9 ± 111.7	$\mathbf{2290.8 \pm 22.5}$	1897.9 ± 63.6	$\mathbf{2087.7 \pm 12.1}$
Grids30	4661.2 ± 54.0	$\mathbf{5078.7 \pm 31.8}$	3784.3 ± 318.0	$\mathbf{4606.0 \pm 42.0}$	3230.5 ± 558.9	$\mathbf{4264.6 \pm 24.6}$

CHAPTER 9

CONCLUSION

In this dissertation, we proposed a novel variational inference method for solving inference problems in probabilistic graphical models. The proposed method is robust to loopy graphs and can handle mixed case with continuous and discrete variables. The method has many advantages comparing to both message passing based methods and other variational inference algorithms. Our method does not suffer from the convergence issues that can limit the practical performance of particle message-passing approaches, and it scales well to large size problems. We demonstrate the performance of our method on a variety of different inference tasks. And the proposed method outperforms the state-of-the-art methods both in terms of speed and accuracy. There are several limitations as well as possible extensions for our method. First, our method can not work in the case when BFE is unbounded from above. BFE can be unbounded either with unbounded potential functions or with unbounded entropy of beliefs and extra efforts are needed for bounding BFE in this case. Second, our method can be applied to inference tasks with new kinds of potential functions, e.g. neural networks. Third, to reach beyond inference problems, our method can also be used as a subroutine for learning.

APPENDIX A

STOCHASTIC EXPECTATION DERIVATIVE CALCULATION FOR LDA MODEL

First let us define a general form which can describe all the expectation terms in (7.6). Let $p(\tau)$ be a probability distribution and $f(X,\tau)$ be a function. Assume τ is the set of parameters of which we want to calculate their derivatives and X represents all the other parameters which can be regarded as constants.

$$\begin{aligned}
\nabla_\tau \mathbb{E}_{p(\tau)}[f(X,\tau)] &= \int_X \nabla_\tau [p(\tau) f(X,\tau)] dX \\
&= \int_X \nabla_\tau p(\tau) f(X,\tau) + p(\tau) \nabla_\tau f(X,\tau) dX \\
&= \int_X \frac{\nabla_\tau p(\tau)}{p(\tau)} p(\tau) f(X,\tau) dX + \int_X p(\tau) \nabla_\tau f(X,\tau) dX \\
&= \int_X p(\tau) f(X,\tau) \nabla_\tau \log p(\tau) dX + \int_X p(\tau) \nabla_\tau f(X,\tau) dX \\
&= \mathbb{E}_{p(\tau)}[f(X,\tau) \nabla_\tau \log p(\tau)] + \mathbb{E}_{p(\tau)}[\nabla_\tau f(X,\tau)] \\
&= \mathbb{E}_{p(\tau)}[f(X,\tau) \nabla_\tau \log p(\tau) + \nabla_\tau f(X,\tau)] \qquad \text{(A.1)}
\end{aligned}$$

Given the BFE in (7.6), we care about the derivatives with respect to the parameters $\alpha^m_{\beta_k}$, $\alpha^m_{\theta_d}$, $\pi^m_{w_{d,n}}(z)$ and the mixture weights λ_m with all the other variables fixed. Note that $\alpha^m_{\beta_k}$ and $\alpha^m_{\theta_d}$ are vectors here. And the derivative of the gamma function can be described in terms of $\Gamma'(x) = \Gamma(x)\psi_0(x)$ where $\psi_0(x)$ is polygamma function. As for this stochastic gradient ascent problem, we first sample a random document d and apply Eq. A.1 to the corresponding terms:

$$\begin{aligned}
\nabla_{\pi^m_{w_{d,n}}(z)} F(b) =& \lambda_m \Big[(\log b_{w_{d,n}}(z) + 1) \\
&+ \mathbb{E}_{\prod_{k=1}^K Dir(\beta_k | \alpha^m_{\beta_k})} [\log p(w_{d,n} | z_{d,n}, \beta) - (\log b_{w_{d,n}}(z, \beta) + 1)] \\
&+ \mathbb{E}_{Dir(\theta_d | \alpha^m_{\theta_d})} [\log p(z_{d,n} | \theta_d) - (\log b_{w_{d,n}}(z, \theta_d) + 1)] \Big] \qquad \text{(A.2)}
\end{aligned}$$

$$
\begin{aligned}
\nabla_{\{\alpha^m_{\beta_k}\}_i} F(b) =& \lambda_m \Bigg[\mathbb{E}_{Dir(\beta_k|\alpha^m_{\beta_k})} \Big[[\log p(\beta_k|\eta) - (\log b_{\beta_k}(\beta_k) + 1)] \\
& [\psi(\sum_{j=1}^{V} \{\alpha^m_{\beta_k}\}_j) - \psi(\{\alpha^m_{\beta_k}\}_i) + \log(\{\beta_k\}_i)] \Big] \\
& + \sum_{n=1}^{N_d} \mathbb{E}_{\prod_{k=1}^{K} Dir(\beta_k|\alpha^m_{\beta_k})} \Big[[\log b_\beta(\beta) + 1] \\
& [\psi(\sum_{j=1}^{V} \{\alpha^m_{\beta_k}\}_j) - \psi(\{\alpha^m_{\beta_k}\}_i) + \log(\{\beta_k\}_i)] \Big] \\
& + \sum_{n=1}^{N_d} \sum_{z=1}^{K} \mathbb{E}_{\prod_{k=1}^{K} Dir(\beta_k|\alpha^m_{\beta_k})} \Big[[\log p(w_{d,n}|z_{d,n}, \beta) - (\log b_{w_{d,n}}(z, \beta) + 1)] \\
& \pi^m_{w_{d,n}}(z) [\psi(\sum_{j=1}^{V} \{\alpha^m_{\beta_k}\}_j) - \psi(\{\alpha^m_{\beta_k}\}_i) + \log(\{\beta_k\}_i)] \Big] \Bigg]
\end{aligned} \tag{A.3}
$$

$$
\begin{aligned}
\nabla_{\{\alpha^m_{\theta_d}\}_i} F(b) =& \lambda_m \Bigg[\mathbb{E}_{Dir(\theta_d|\alpha^m_{\theta_d})} \Big[[\log p(\theta_d|\gamma) - (1 - N_d)(\log b_{\theta_d}(\theta_d) + 1)] \\
& [\psi(\sum_{j=1}^{K} \{\alpha^m_{\theta_d}\}_j) - \psi(\{\alpha^m_{\theta_d}\}_i) + \log(\{\theta_d\}_i)] \Big] \\
& + \sum_{n=1}^{N_d} \sum_{z=1}^{K} \mathbb{E}_{Dir(\theta_d|\alpha^m_{\theta_d})} \Big[[\log p(z_{d,n}|\theta_d) - \log b_{w_{d,n}}(z, \theta_d) - 1] \\
& \pi^m_{w_{d,n}}(z) [\psi(\sum_{j=1}^{K} \{\alpha^m_{\theta_d}\}_j) - \psi(\{\alpha^m_{\theta_d}\}_i) + \log(\{\theta_d\}_i)] \Big] \Bigg]
\end{aligned} \tag{A.4}
$$

$$
\begin{aligned}
\nabla_{\lambda_m} F(b) =& \sum_{n=1}^{N_d} \sum_{z=1}^{K} \pi^m_{w_{d,n}}(z) (\log b_{w_{d,n}}(z) + 1) \\
& + \sum_{k=1}^{K} \mathbb{E}_{Dir(\beta_k|\alpha^m_{\beta_k})} [\log p(\beta_k|\eta) - (\log b_{\beta_k}(\beta_k) + 1)] \\
& + \mathbb{E}_{Dir(\theta_d|\alpha^m_{\theta_d})} [\log p(\theta_d|\gamma) - (1 - N_d)(\log b_{\theta_d}(\theta_d) + 1)] \\
& + \sum_{n=1}^{N_d} \mathbb{E}_{\prod_{k=1}^{K} Dir(\beta_k|\alpha^m_{\beta_k})} \Big[\log b_\beta(\beta) + 1 \Big]
\end{aligned}
$$

$$+\sum_{n=1}^{N_d}\sum_{z=1}^{K}\mathbb{E}_{\prod_{k=1}^{K} Dir(\beta_k|\alpha_{\beta_k}^m)}\Big[[\log p(w_{d,n}|z_{d,n},\beta)-(\log b_{w_{d,n}}(z,\beta)+1)]\pi_{w_{d,n}}^m(z)\Big]$$
$$+\sum_{n=1}^{N_d}\sum_{z=1}^{K}\mathbb{E}_{Dir(\theta_d|\alpha_{\theta_d}^m)}\Big[[\log p(z_{d,n}|\theta_d)-(\log b_{w_{d,n}}(z,\theta_d)+1)]\pi_{w_{d,n}}^m(z)\Big] \tag{A.5}$$

Considering the discrete distribution $\pi_{w_{d,n}}^m(z)$ and mixture weights λ_m may not be a valid probability distributions after the gradient ascent update, we apply a change of variables to them:

$$\pi_{w_{d,n}}^m(z)=\frac{\exp\left(s_{w_{d,n}}^m(z)\right)}{\sum_z \exp\left(s_{w_{d,n}}^m(z)\right)}$$
$$\lambda_m=\frac{\exp\left(t_m\right)}{\sum_m \exp\left(t_m\right)} \tag{A.6}$$

The gradient ascent process is conducted on the parameters $\alpha_{\beta_k}^m$, $\alpha_{\theta_d}^m$, $s_{w_{d,n}}^m(z)$, t_m and γ_k.

APPENDIX B

HANDLE LOG-BELIEFS WITH HIGH DIMENSIONAL DIRICHLET DISTRIBUTION

The calculation of the derivatives require evaluation of sums of high(thousands) dimensional Dirichlet distributions that are very close to zero, which results in zeros when evaluated by the machines. Take log-belief term $\log b_\beta(\beta_k)$ as an example, we can transform it into another form.

$$\log b_\beta(\beta_k) = \log \sum_{m=1}^{M} \lambda_m Dir(\beta_k|\alpha_{\beta_k}^m) = \log \sum_{m=1}^{M} \frac{\lambda_m Dir(\beta_k|\alpha_{\beta_k}^m)}{\lambda_{m^*} Dir(\beta_k|\alpha_{\beta_k}^{m^*})} + \log \lambda_{m^*} Dir(\beta_k|\alpha_{\beta_k}^{m^*}) \tag{B.1}$$

where $m^* = \max_m\{\log \lambda_m Dir(\beta_k|\alpha_{\beta_k}^m)\}$.

In this case with $\log 0 = 0$ defined, we only need to evaluate log-Dirichlet function which is doable for the machines. And we can pre-compute all the log-Dirichlet values as well.

APPENDIX C

SVGD $\phi^*(x_i)$ FUNCTION DERIVATION

Consider $p(x) = b_\beta(\beta)$ and $f(x) = b_\beta(\beta)$. We use RBF kernel function with $\sigma = 1$ and $x_i = \beta_i$ is a vocabulary size vector of particles. The $\phi^*(x_i)$ function can be written as:

$$\phi^*(x_i) = \phi^*(\begin{bmatrix} \beta_{i1} \\ \vdots \\ \beta_{iV} \end{bmatrix}) = \frac{1}{N} \sum_{j=1}^{N} \Big[\frac{1}{b_{\beta(\beta_j)}} \begin{bmatrix} \sum_{m=1}^{M} \frac{\lambda_m(\alpha_{\beta_1}^m - 1)}{\beta_{j1}} Dir(\beta_j | \alpha_\beta^m) \\ \vdots \\ \sum_{m=1}^{M} \frac{\lambda_m(\alpha_{\beta_V}^m - 1)}{\beta_{jV}} Dir(\beta_j | \alpha_\beta^m) \end{bmatrix} \cdot \begin{bmatrix} e^{-\frac{(\beta_{j1} - \beta_{i1})^2}{2}} \\ \vdots \\ e^{-\frac{(\beta_{jV} - \beta_{iV})^2}{2}} \end{bmatrix} - \begin{bmatrix} e^{-\frac{(\beta_{j1} - \beta_{i1})^2}{2}} (\beta_{j1} - \beta_{i1}) \\ \vdots \\ e^{-\frac{(\beta_{jV} - \beta_{iV})^2}{2}} (\beta_{jV} - \beta_{iV}) \end{bmatrix} \Big] \tag{C.1}$$

The derivation of $\phi^*(x_i^t)$ function of the expectation terms in the BFE are similar to Eq. C.1.

APPENDIX D

NONPARAMETRIC VARIATIONAL INFERENCE DERIVATIVES

In LDA model for one sampled document d, the belief under the nonparametric variational inference assumption can be defined as:

$$b(z, X_d) = \sum_m \lambda_m \prod_i \pi^m_{w_{d,n}}(z_i) Dir(x_{\theta_d}|\alpha^m_{\theta_d}) \prod_k Dir(x_{\beta_k}|\alpha^m_{\beta_k}) \tag{D.1}$$

The entropy part of BFE with the above beliefs is:

$$\begin{aligned} -\sum_z \int_{X_d} b(z, X_d) \log b(z, X_d) \geq & -\sum_{m_2} \lambda_{m_2} \log \big[\sum_z \int_{X_d} b(z, X_d) \prod_i \pi^{m_2}_{w_{d,n}}(z_i) \\ & Dir(x_{\theta_d}|\alpha^{m_2}_{\theta_d}) \prod_k Dir(x_{\beta_k}|\alpha^{m_2}_{\beta_k}) \big] \end{aligned} \tag{D.2}$$

Define $\square_{m_2}$ to be in the form of

$$\begin{aligned} \square_{m_2} =& \sum_z \int_{X_d} b(z, X_d) \prod_i \pi^{m_2}_{w_{d,n}}(z_i) Dir(x_{\theta_d}|\alpha^{m_2}_{\theta_d}) \prod_k Dir(x_{\beta_k}|\alpha^{m_2}_{\beta_k}) \\ =& \sum_{m_1} \lambda_{m_1} \prod_i \big(\sum_z \pi^{m_1}_i(z) \pi^{m_2}_i(z) \big) \mathbb{E}_{Dir(\alpha^{m_2}_{\theta_d}) \prod_k Dir(\alpha^{m_2}_{\beta_k})} \big[Dir(\alpha^{m_1}_{\theta_d}) \prod_k Dir(\alpha^{m_1}_{\beta_k}) \big] \end{aligned} \tag{D.3}$$

The derivatives with respect to the parameters can be calculated as:

$$\begin{aligned} \nabla_{\pi^{m_1}_{w_{d,n}}(z)} =& \lambda_{m_1} \mathbb{E}_{\prod_{k=1}^K Dir(\beta_k|\alpha^m_{\beta_k})} \big[\log p(w_{d,n}|z, \beta) \big] \\ & + \lambda_{m_1} \mathbb{E}_{Dir(x_{\theta_d}|\alpha^{m_1}_{\theta_d})} \big[\log p(z|\theta_d) \big] \\ & - \sum_{m_2 \neq m_1} \lambda_{m_2} \frac{1}{\square_{m_2}} \Big[\lambda_{m_1} \prod_{i \neq w_{d,n}} \big(\sum_z \pi^{m_1}_i(z) \pi^{m_2}_i(z) \big) \\ & \mathbb{E}_{Dir(\alpha^{m_2}_{\theta_d}) \prod_k Dir(\alpha^{m_2}_{\beta_k})} \big[Dir(\alpha^{m_1}_{\theta_d}) \prod_k Dir(\alpha^{m_1}_{\beta_k}) \big] \Big] \\ & - \lambda_{m_1} \frac{1}{\square_{m_1}} \Big[\sum_{m_2} \lambda_{m_2} \prod_{i \neq w_{d,n}} \big(\sum_z \pi^{m_1}_i(z) \pi^{m_2}_i(z) \big) \\ & \mathbb{E}_{Dir(\alpha^{m_2}_{\theta_d}) \prod_k Dir(\alpha^{m_2}_{\beta_k})} \big[Dir(\alpha^{m_1}_{\theta_d}) \prod_k Dir(\alpha^{m_1}_{\beta_k}) \big] \Big] \end{aligned} \tag{D.4}$$

$$
\begin{aligned}
\nabla_{\{\alpha_{\theta_d}^{m_1}\}_i} =& \lambda_{m_1}\mathbb{E}_{Dir(x_{\theta_d}|\alpha_{\theta_d}^{m_1})}\Big[\log p(x_{\theta_d}|\gamma)[\psi(\sum_{j=1}^{K}\{\alpha_{\theta_d}^{m_1}\}_j)-\psi(\{\alpha_{\theta_d}^{m_1}\}_i)+\log(\{x_{\theta_d}\}_i)]\Big] \\
&+\lambda_{m_1}\sum_{n=1}^{N_d}\sum_{z=1}^{K}\mathbb{E}_{Dir(x_{\theta_d}|\alpha_{\theta_d}^{m_1})}\Big[\log p(z_{d,n}|x_{\theta_d})\pi_{w_{d,n}}^{m_1}(z) \\
&\quad [\psi(\sum_{j=1}^{K}\{\alpha_{\theta_d}^{m_1}\}_j)-\psi(\{\alpha_{\theta_d}^{m_1}\}_i)+\log(\{x_{\theta_d}\}_i)]\Big] \\
&-\sum_{m_2\neq m_1}\lambda_{m_2}\frac{1}{\square_{m_2}}\Big[\lambda_{m_1}\prod_i\big(\sum_z \pi_i^{m_1}(z)\pi_i^{m_2}(z)\big) \\
&\quad \mathbb{E}_{Dir(\alpha_{\theta_d}^{m_2})\prod_k Dir(\alpha_{\beta_k}^{m_2})}\big[Dir(\alpha_{\theta_d}^{m_1})\prod_k Dir(\alpha_{\beta_k}^{m_1}) \\
&\quad [\psi(\sum_{j=1}^{K}\{\alpha_{\theta_d}^{m_1}\}_j)-\psi(\{\alpha_{\theta_d}^{m_1}\}_i)+\log(\{x_{\theta_d}\}_i)]\big]\Big] \\
&-\lambda_{m_1}\frac{1}{\square_{m_1}}\Big[\sum_{m_2}\lambda_{m_2}\prod_i\big(\sum_z \pi_i^{m_1}(z)\pi_i^{m_2}(z)\big) \\
&\quad \mathbb{E}_{Dir(\alpha_{\theta_d}^{m_2})\prod_k Dir(\alpha_{\beta_k}^{m_2})}\big[Dir(\alpha_{\theta_d}^{m_1})\prod_k Dir(\alpha_{\beta_k}^{m_1}) \\
&\quad [\psi(\sum_{j=1}^{K}\{\alpha_{\theta_d}^{m_1}\}_j)-\psi(\{\alpha_{\theta_d}^{m_1}\}_i)+\log(\{x_{\theta_d}\}_i)]\big]\Big]
\end{aligned}
\tag{D.5}
$$

$$
\begin{aligned}
\nabla_{\{\alpha_{\beta_k}^{m_1}\}_i} =& \lambda_{m_1}\mathbb{E}_{Dir(x_{\beta_k}|\alpha_{\beta_k}^{m_1})}\Big[\log p(x_{\beta_k}|\eta)[\psi(\sum_{j=1}^{K}\{\alpha_{\beta_k}^{m_1}\}_j)-\psi(\{\alpha_{\beta_k}^{m_1}\}_i)+\log(\{x_{\beta_k}\}_i)]\Big] \\
&+\lambda_{m_1}\sum_{n=1}^{N_d}\sum_{z=1}^{K}\mathbb{E}_{\prod_{k=1}^{K} Dir(x_{\beta_k}|\alpha_{\beta_k}^{m_1})}\Big[\log p(w_{d,n}|z_{d,n},x_\beta)\pi_{w_{d,n}}^{m_1}(z) \\
&\quad [\psi(\sum_{j=1}^{K}\{\alpha_{\beta_k}^{m_1}\}_j)-\psi(\{\alpha_{\beta_k}^{m_1}\}_i)+\log(\{x_{\beta_k}\}_i)]\Big] \\
&-\sum_{m_2\neq m_1}\lambda_{m_2}\frac{1}{\square_{m_2}}\Big[\lambda_{m_1}\prod_i\big(\sum_z \pi_i^{m_1}(z)\pi_i^{m_2}(z)\big) \\
&\quad \mathbb{E}_{Dir(\alpha_{\theta_d}^{m_2})\prod_k Dir(\alpha_{\beta_k}^{m_2})}\big[Dir(\alpha_{\theta_d}^{m_1})\prod_k Dir(\alpha_{\beta_k}^{m_1}) \\
&\quad [\psi(\sum_{j=1}^{K}\{\alpha_{\beta_k}^{m_1}\}_j)-\psi(\{\alpha_{\beta_k}^{m_1}\}_i)+\log(\{x_{\beta_k}\}_i)]\big]\Big]
\end{aligned}
$$

$$
\begin{aligned}
&-\lambda_{m_1}\frac{1}{\square_{m_1}}\Big[\sum_{m_2}\lambda_{m_2}\prod_i\big(\sum_z \pi_i^{m_1}(z)\pi_i^{m_2}(z)\big)\\
&\mathbb{E}_{Dir(\alpha_{\theta_d}^{m_2})\prod_k Dir(\alpha_{\beta_k}^{m_2})}\big[Dir(\alpha_{\theta_d}^{m_1})\prod_k Dir(\alpha_{\beta_k}^{m_1})\\
&[\psi(\sum_{j=1}^{K}\{\alpha_{\beta_k}^{m_1}\}_j)-\psi(\{\alpha_{\beta_k}^{m_1}\}_i)+\log(\{x_{\beta_k}\}_i)]\big]\Big]
\end{aligned}
\tag{D.6}
$$

REFERENCES

Anderson, J. R. and C. Peterson (1987). A mean field theory learning algorithm for neural networks. *Complex Systems 1*, 995–1019.

Aronszajn, N. (1950). Theory of reproducing kernels. *Transactions of the American mathematical society 68*(3), 337–404.

Barber, D. and W. Wiegerinck (1999). Tractable variational structures for approximating graphical models. In *Advances in Neural Information Processing Systems*, pp. 183–189.

Belanger, D., D. Sheldon, and A. McCallum (2013). Marginal inference in mrfs using frank-wolfe. In *NIPS Workshop on Greedy Optimization, Frank-Wolfe and Friends*.

Bickel, P. J. and K. A. Doksum (2015). *Mathematical statistics: basic ideas and selected topics, volume I*, Volume 117. CRC Press.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. springer.

Bishop, C. M., N. D. Lawrence, T. Jaakkola, and M. I. Jordan (1998). Approximating posterior distributions in belief networks using mixtures. In *Advances in neural information processing systems*, pp. 416–422.

Blei, D. M., A. Kucukelbir, and J. D. McAuliffe (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association 112*(518), 859–877.

Blei, D. M., A. Y. Ng, and M. I. Jordan (2003). Latent dirichlet allocation. *Journal of machine Learning research 3*(Jan), 993–1022.

Chib, S. and B. P. Carlin (1999). On mcmc sampling in hierarchical longitudinal models. *Statistics and Computing 9*(1), 17–26.

Chow, C. and C. Liu (1968). Approximating discrete probability distributions with dependence trees. *IEEE transactions on Information Theory 14*(3), 462–467.

Cooper, G. F. (1990). The computational complexity of probabilistic inference using bayesian belief networks. *Artificial intelligence 42*(2-3), 393–405.

Cseke, B. and T. Heskes (2011). Properties of Bethe free energies and message passing in Gaussian models. *Journal of Artificial Intelligence Research*, 1–24.

Darwiche, A. (2009). *Modeling and reasoning with Bayesian networks*. Cambridge university press.

Dheeru, D. and E. Karra Taniskidou (2017). UCI machine learning repository.

Fleet, D. and Y. Weiss (2006a). Optical flow estimation. In *Handbook of mathematical models in computer vision*, pp. 237–257. Springer.

Fleet, D. and Y. Weiss (2006b). Optical flow estimation. In *Handbook of mathematical models in computer vision*, pp. 237–257. Springer.

Frank, A., P. Smyth, and A. T. Ihler (2009). Particle-based variational inference for continuous systems. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 826–834.

Geiger, D. and D. Heckerman (1996). Knowledge representation and inference in similarity networks and bayesian multinets. *Artificial Intelligence 82*(1-2), 45–74.

Gershman, S. J., M. D. Hoffman, and D. M. Blei (2012). Nonparametric variational inference. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pp. 235–242. Omnipress.

Globerson, A. and T. S. Jaakkola (2008). Fixing max-product: Convergent message passing algorithms for map lp-relaxations. In *Advances in neural information processing systems*, pp. 553–560.

Golub, G. H. and J. H. Welsch (1969). Calculation of Gauss quadrature rules. *Mathematics of computation 23*(106), 221–230.

Halin, R. (1976). S-functions for graphs. *Journal of geometry 8*(1-2), 171–186.

Heckerman, D. (2008). A tutorial on learning with bayesian networks. In *Innovations in Bayesian networks*, pp. 33–82. Springer.

Heckerman, D. E. and B. N. Nathwani (1992). An evaluation of the diagnostic accuracy of pathfinder. *Computers and Biomedical Research 25*(1), 56–74.

Huber, M. F., T. Bailey, H. Durrant-Whyte, and U. D. Hanebeck (2008). On entropy approximation for Gaussian mixture random vectors. In *Multisensor Fusion and Integration for Intelligent Systems (MFI), IEEE International Conference on*, pp. 181–188.

Ihler, A. T., J. W. Fisher, R. L. Moses, and A. S. Willsky (2005). Nonparametric belief propagation for self-localization of sensor networks. *IEEE Journal on Selected Areas in Communications 23*(4), 809–819.

Ihler, A. T. and D. A. McAllester (2009). Particle belief propagation. In *Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 256–263.

Jaakkola, T. S. and M. I. Jordan (1998). Improving the mean field approximation via the use of mixture distributions. In *Learning in graphical models*, pp. 163–173. Springer.

Jordan, M. I., Z. Ghahramani, T. S. Jaakkola, and L. K. Saul (1999). An introduction to variational methods for graphical models. *Machine Learning 37*(2), 183–233.

Koller, D. and N. Friedman (2009). *Probabilistic graphical models: principles and techniques*. MIT press.

Kolmogorov, V. and M. Wainwright (2012). On the optimality of tree-reweighted max-product message-passing. *arXiv preprint arXiv:1207.1395*.

Kullback, S. and R. A. Leibler (1951). On information and sufficiency. *The annals of mathematical statistics 22*(1), 79–86.

Kumar, M. and D. Koller (2009). Learning a small mixture of trees. *Advances in neural information processing systems 22*, 1051–1059.

Lafferty, J., A. McCallum, and F. C. Pereira (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Lienart, T., Y. W. Teh, and A. Doucet (2015). Expectation particle belief propagation. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 3609–3617.

Liu, Q. and A. Ihler (2013). Variational algorithms for marginal map. *The Journal of Machine Learning Research 14*(1), 3165–3200.

Loeliger, H.-A. (2004). An introduction to factor graphs. *IEEE Signal Processing Magazine 21*(1), 28–41.

London, B., B. Huang, and L. Getoor (2015). The benefits of learning with strongly convex approximate inference. In *International Conference on Machine Learning*, pp. 410–418.

Marinescu, R., R. Dechter, and A. T. Ihler (2014). And/or search for marginal map. In *UAI*, pp. 563–572. Citeseer.

Marinescu, R., J. Lee, A. T. Ihler, and R. Dechter (2017). Anytime best+ depth-first search for bounding marginal map. In *AAAI*, pp. 3775–3782.

Meila, M. and M. I. Jordan (2000). Learning with mixtures of trees. *Journal of Machine Learning Research 1*(Oct), 1–48.

Meltzer, T., A. Globerson, and Y. Weiss (2009). Convergent message passing algorithms: a unifying view. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pp. 393–401. AUAI Press.

Minka, T. P. (2001). Expectation propagation for approximate Bayesian inference. In *Proceedings of the Seventeenth conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 362–369.

Minka, T. P. (2013). Expectation propagation for approximate bayesian inference. *arXiv preprint arXiv:1301.2294*.

Murphy, K., Y. Weiss, and M. I. Jordan (2013). Loopy belief propagation for approximate inference: An empirical study. *arXiv preprint arXiv:1301.6725*.

Murphy, K. P., Y. Weiss, and M. I. Jordan (1999). Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pp. 467–475. Morgan Kaufmann Publishers Inc.

Ni, X., N. Quadrianto, Y. Wang, and C. Chen (2017). Composing tree graphical models with persistent homology features for clustering mixed-type data. In *International Conference on Machine Learning (ICML)*, pp. 2622–2631.

Park, J. D. (2002). Map complexity results and approximation methods. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pp. 388–396. Morgan Kaufmann Publishers Inc.

Pearl, J. (1982). *Reverend Bayes on inference engines: A distributed hierarchical approach*. Cognitive Systems Laboratory, School of Engineering and Applied Science .

Pearl, J. (1985). Bayesian networks: A model of self-activated memory for evidential reasoning. In *Proceedings of the 7th Conference of the Cognitive Science Society, University of California, Irvine, CA, USA*, pp. 15–17.

Pearl, J. (2014). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier.

Pearl, J. and A. Paz (1985). *Graphoids: A graph-based logic for reasoning about relevance relations*. University of California (Los Angeles). Computer Science Department.

Rayana, S. and L. Akoglu (2015). Collective opinion spam detection: Bridging review networks and metadata. In *Proceedings of the 21th acm sigkdd international conference on knowledge discovery and data mining*, pp. 985–994.

Ross Kindermann, J. L. S. (1980). *Markov Random Fields and Their Applications*. AMS.

Roth, D. (1996). On the hardness of approximate reasoning. *Artificial Intelligence 82*(1-2), 273–302.

Schölkopf, B., A. J. Smola, F. Bach, et al. (2002). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

Scott, D. W. (1992). *Multivariate density estimation: theory, practice, and visualization*. Wiley.

Shawe-Taylor, J., N. Cristianini, et al. (2004). *Kernel methods for pattern analysis*. Cambridge university press.

Song, L., A. Gretton, D. Bickson, Y. Low, and C. Guestrin (2011). Kernel belief propagation. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 707–715.

Stratonovich, R. L. (1965). Conditional markov processes. In *Non-linear transformations of stochastic processes*, pp. 427–453. Elsevier.

Sudderth, E. B., A. T. Ihler, W. T. Freeman, and A. S. Willsky (2003). Nonparametric belief propagation. In *Proceedings of the 2003 IEEE computer society conference on Computer vision and pattern recognition*, pp. 605–612. IEEE Computer Society.

Szeliski, R., R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother (2008). A comparative study of energy minimization methods for markov random fields with smoothness-based priors. *IEEE transactions on pattern analysis and machine intelligence 30*(6), 1068–1080.

Taga, N. and S. Mase (2006). On the convergence of loopy belief propagation algorithm for different update rules. *IEICE transactions on fundamentals of electronics, communications and computer sciences 89*(2), 575–582.

Vrontos, I. D., P. Dellaportas, and D. N. Politis (2000). Full bayesian inference for garch and egarch models. *Journal of Business & Economic Statistics 18*(2), 187–198.

Wainwright, M. J., T. S. Jaakkola, and A. S. Willsky (2003). Tree-reweighted belief propagation algorithms and approximate ml estimation by pseudo-moment matching. In *AISTATS*, Volume 3, pp. 3.

Wainwright, M. J., T. S. Jaakkola, and A. S. Willsky (2005, Nov.). MAP estimation via agreement on (hyper)trees: Message-passing and linear programming. *Information Theory, IEEE Transactions on 51*(11), 3697–3717.

Wainwright, M. J. and M. I. Jordan (2008). *Graphical models, exponential families, and variational inference*. Now Publishers Inc.

Wang, D., Z. Zeng, and Q. Liu (2018). Stein variational message passing for continuous graphical models. In *International Conference on Machine Learning (ICML)*.

Weiss, Y. and W. T. Freeman (2001). Correctness of belief propagation in Gaussian graphical models of arbitrary topology. *Neural computation 13*(10), 2173–2200.

Werner, T. (2007). A linear programming approach to max-sum problem: A review. *IEEE transactions on pattern analysis and machine intelligence 29*(7), 1165–1179.

Xue, Y., Z. Li, S. Ermon, C. P. Gomes, and B. Selman (2016). Solving marginal map problems with np oracles and parity constraints. In *Advances in Neural Information Processing Systems*, pp. 1127–1135.

Yedidia, J. S., W. T. Freeman, and Y. Weiss (2005a). Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on information theory 51*(7), 2282–2312.

Yedidia, J. S., W. T. Freeman, and Y. Weiss (2005b, July). Constructing free-energy approximations and generalized belief propagation algorithms. *Information Theory, IEEE Transactions on 51*(7), 2282 – 2312.

Zhang, N. L. and D. Poole (1994). A simple approach to bayesian network computations. In *Proc. of the Tenth Canadian Conference on Artificial Intelligence*.

Zweig, G. and S. Russell (1998). Speech recognition with dynamic bayesian networks.

BIOGRAPHICAL SKETCH

Yuanzhen Guo earned his Bachelors degree in 2013 majored in Computer Science and Technology from Harbin Institute of Technology, China. In 2015, he completed his Masters degree from Waseda University, Japan majored in Computer Engineering. He started his PhD in Fall 2015 at the University of Texas at Dallas serving as a teaching/research assistant during his study.

CURRICULUM VITAE

Yuanzhen Guo

November 12, 2020

Contact Information

Department of Computer Science
The University of Texas at Dallas
800 W. Campbell Rd.
Richardson, TX 75080-3021, U.S.A.

Email: yuanzhen.guo@utdallas.edu

Educational History

B.S., Computer Science and Technology, Harbin Institute of Technology, 2013

M.S., Computer Engineering, Waseda University, 2015

Ph.D., Computer Science, The University of Texas at Dallas, 2020

Variational Inference Methods for Continuous Probabilistic Graphical Models
Ph.D. Dissertation
Computer Science Department, The University of Texas at Dallas
Advisor: Dr. Nicholas Ruozzi

Detection of Text-based Advertising and Promotion in Wikipedia by Deep Learning Methods
Master Thesis
Graduate School of Information, Production and Systems, Waseda University
Advisor: Dr. Mizuho Iwaihara

Employment History

Applied Scientist Intern, Amazon, May 2020 - Aug 2020

Data Scientist Intern, Insight Enterprises, May 2019 - Aug 2019

Research graduate student, Los Alamos National Lab, May 2017 - Aug 2017