Perspectives on Probabilistic Graphical Models

DONG LIU

Doctoral Thesis in Electrical Engineering
Stockholm, Sweden 2020

Devision of Information Science and Engineering
TRITA-EE XXXX KTH, School of Electrical Engineering and and Computer Science
ISSN .... SE-100 44 Stockholm
ISBN .... SWEDEN

Akademisk avhandling som med tillstånd av Kungl Tekniska högskolan framlägges till offentlig granskning för avläggande av teknologie doktorsexamen i Elektroteknik onsdag den 13 september 2017 klockan 13.15 i F3, Lindstedtsvägen 26, Stockholm.

Tryck: Universitetsservice US AB

To my beloved

Abstract

Sammanfattning

Acknowledgements

Contents

Acronyms and Notations

Notations

X	random variable
x	realization of the random variable X
$\mathcal{X}$	alphabet of the random variable X
X_i^k	random sequence $(X_i, \dots, X_k)$
x_i^k	realization of the random sequence X_i^k
$\mathcal{X}_i^k$	alphabet of the random sequence X_i^k
X^k	random sequence $(X_1, \dots, X_k)$
x^k	realization of the random sequence X^k
$\mathcal{X}^k$	alphabet of the random sequence X^k
$X_i^{k\backslash n}$	random sequence $(X_i, \dots, X_{n-1}, X_{n+1}, \dots, X_k)$
$x_i^{k\backslash n}$	realization of the random sequence $X_i^{k\backslash n}$
$\mathcal{X}_i^{k\backslash n}$	alphabet of the random sequence $X_i^{k\backslash n}$
$X^{k\backslash n}$	random sequence $(X_1, \dots, X_{n-1}, X_{n+1}, \dots, X_k)$
$x^{k\backslash n}$	realization of the random sequence $X^{k\backslash n}$
$\mathcal{X}^{k\backslash n}$	alphabet of the random sequence $X^{k\backslash n}$
$\|\| \cdot \|\|$	set cardinality
f_X	p.d.f. of the continuous random variable X
p_X	p.m.f. of the discrete random variable X
$\mathcal{N}(\mu, \sigma^2)$	normal distribution with mean μ and variance σ^2

$\mathrm{D}(\cdot\|\|\cdot)$	Kullback-Leibler divergence
$\mathrm{D}_\tau(\cdot\|\|\cdot)$	τ-th order Rényi divergence
$\mathrm{C}(\cdot,\cdot)$	Chernoff information
$\mathrm{E}[\cdot]$	expectation
$\partial\cdot$	boundary of a closed set
$\hat{\partial}\cdot$	upper boundary of a two-dimensional closed set
$\check{\partial}\cdot$	lower boundary of a two-dimensional closed set
$\log(\cdot)$	natural logarithm

Chapter 1

Introduction

1.1 Motivations

Most tasks conducted by a person or an automated system requires a fundamental ability of *reasoning*, which is always about reaching a conclusion based on available information. At times, a conclusion is not enough and it is also required to know how reliable the conclusion is. Take the coronavirus (COVID-19) that started from Wuhan in China at the end of 2019 as example, a doctor needs to check the information about a person to reason if the person is infected by the coronavirus. The relevant information includes symptoms such as fever, cough, breathing difficulties and probably kidney failure in severe cases. Even after the doctor has reached a conclusion of positive or negative infection of coronavirus for the person, a natural question is why and how *confident* the diagnose is made.

Instead of randomly guessing, reasoning is to answer queries with preserving uncertainty, by making best of available information. Two fundamental problems are inevitable before a rational reasoning can be conducted.

- How should we specify the relationship between a conclusion and the available information? In the coronavirus example, the counterpart question to answer is how the doctor should relate coronavirus infection with the symptoms. This step is called *modeling* which represents a reasoning problem abstractly by specifying the relationship between known information and unknown parts, in preparation of answer query on it.

- With the modeled representation, how a conclusion should be made? This process of reaching a answer to the query from an abstracted representation is called *inference*. Assume the doctor knows that the candidate has fever and cough, but not any breathing problem, how likely does the candidate have be infected by coronavirus?

In the modeling and inference process, it is not likely that the very beginning assumptions are perfectly right about the truth and can be used for all instances of the same type of queries. On the contrary, we usually begin with simple (sometimes naive) assumptions of representation a problem, and come back to revise it later when more information or evidence is available, which is aligned with our learning process of new knowledge. In fact, this assumption-and-revision procedure can be more compact. Instead of fixing the model representation at the beginning when one might not be sure the correctness of the assumptions about the model, one can assume a set of models with the assumption that the 'correct' model is in this set. A typical strategy is to leave some freedom to the configuration of the model at the beginning. Each instantiating the configuration generates a model representation. By using available observations or information, the model is adjusted to be able to be best compatible with the observations. This procedure adds the following fundamental problem in reasoning.

- Instead of having a fixed model at the first step, a set of models (or hypothesis) is given. We then need to choose one model that best describes the available observations or information. This phase of choosing a model is called *learning.*

Afterwards, inference can be conducted on the learned model to answer queries.

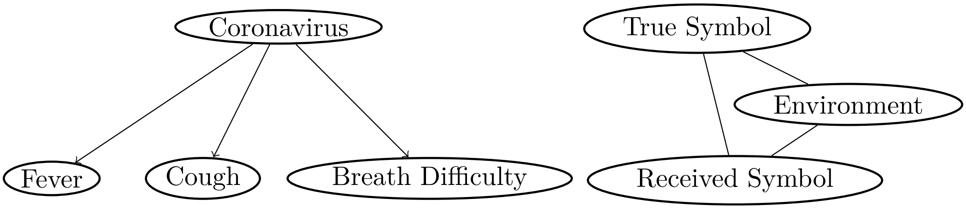
With all the discussed problems above, modeling, inference and learning, our purpose is to carry out reasoning with being aware of how confident a conclusion or answer is. These problems can be treated nicely with probabilistic models via Bayesian logic. Probabilistic model is built on the fundamental calculus of probability theory that is natural to accommodate the *uncertainty*, which is desired in reasoning. In additional, the probabilistic models offers rich space to modeling problems, where inference can be carried on either exactly or approximately. **More importantly, the modeling or modeling learning part is not necessarily coupled with inference algorithms**. This proper separation allows freedom that a certain family of general inference algorithms can be applies to a broad class probabilistic models. It offers the freedom of trying different model representation of a class without the need of replacing inference algorithm.

Example 1. *Consider the coronavirus infection problem. Using probabilistic model, we are able to model the problem in a rigid way. Additionally, we can make query more formally in probabilistic model framework. Assume each symptom among fever, cough and breathing difficulty can take value from* $\{\text{True}, \text{False}\}$*. Also the coronavirus infection is either true or false. One exemplified query can be*

$$P(\text{Infection} = \text{True} | \text{Fever} = \text{True}, \text{Cough} = \text{False}, \text{BreathingDifficulty} = \text{True}),$$

which is asking how likely the patient is infected by coronavirus if symptoms of both fever and breathing difficulty are observed but no sight of cough symptom.

Given the fact that probabilistic theory offers a rigid foundation to model and study the problems, which is used to answer query that we concern, it soon becomes intractable when dozens or hundreds of relevant attributes are joint considered.



(a) A directed graphical model for Example 1.

(b) An undirected graphical model.

Figure 1.1: Different perspectives on probabilistic graphic models. 1.1a A toy Bayesian network. 1.1b A toy Markov random field.

This can be exemplified by giving finer levels of each symptom in coronavirus infection, e.g. symptom fever is represented by the actual body temperature in integer instead of true-or-false binary state on the one hand. On the other hand, there could be more directly and indirectly relevant symptoms such as muscle pain and congestion. Together with the symptoms, the recent travel itinerary is also related. Additionally, season flu could also similarly bring up some symptoms listed above.

Probabilistic graphical model offers a general framework to encode random variable dependency of a complex probabilistic distribution into a structured graph, which is a powerful tool to compactly model relevant attributes and facts of a complex problem or a system. As show in Figure 1.1a that represents the problem of Example 1 into a directed graphical model (also called Bayesian network or generative model), the nodes (or vertexes) correspond to the variables that represents symptoms and infection state, whereas the edges between nodes correspond how one variable may influence others. In certain scenarios, it is natural to use directed graphical models (generative models) to represent that a observable variable is dependent on a latent variable that generates the observations. For instance, a noisy location sensor of a car keeps measuring the car's true location and reading noisy locations.

In contract to the directed graphical model, there are more scenarios that the interaction between related random variables is not directional and an undirected edge is used, which leads to the undirected graphical model (or Markov random field, Section 2.1) representation. Undirected graphical models are popularly used in computer vision, computational biology, digital communication, statistical physics, etc. Figure 1.1b illustrates an exemplified undirected graphical model in digital communication context. On the one hand, a receiver wants to know what is the true symbol by joint considering its communication environment and its received symbol. On the other hand, the symbol received by the receiver is jointly formulated by the true symbol and communication environment. The influence among them is apparently not directional since the impact along an edge can be bidirectional in this example.

Probabilistic graphical model offers a 'scientific language' to do reasoning with

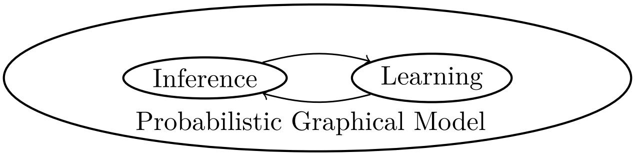


Figure 1.2: Two key aspects in practical graphical models.

uncertainty within framework of probabilistic theory. It is usually a nature representation for a complex system or problem and offers straightforward abstraction. The compact representation of probabilistic graphical model bridges the joint distribution of a complex system, and its graphical abstraction that captures the statistic dependency reflecting our understanding of the system. The advantage of its representation power is one of the reasons that leads to its popularity in difference disciplines.

Probabilistic graphical model coupled with its underlining distribution is a powerful tool for effective inference, apart from its advantage of representation power. It allows to answer queries with the help of the underlining distribution when practical inference algorithms are provided, which meets our need of reasoning with uncertainty. In addition to inference, probabilistic graphical model also supports learning from data. With certain amount of data available, a probabilistic graphical model can be learned to explain the observed data better in addition to align with our own understanding of a domain. The learned graphical model can serve to do inference with higher confidence in return. A diagram is illustrated in Figure 1.2. As would become clear in Part II, the inference may be needed to carry out model learning as well, apart from the above mutual-benefiting interaction.

1.2 Scope and Thesis Outline

We gives the intuition and motivation of probabilistic graphical model in last section, and the interaction between inference and learning in this framework. In this section, we would navigate further among the topics within this framework and states the ones that we would cover in the thesis.

Inference in probabilistic graphical model is about answering queries with help of its coupled distribution. These queries can be generally grouped into the following cases:

- Computing the likelihood of observed data or unobserved random variable.
- Computing the marginals distribution over a particular subset of random variables.
- Computing the conditional distribution a subset of variables given the configuration of another subset of variables.
- Computing the most likely configuration of (a subset of) variables.

The work of this thesis would be mainly related with the first three cases in inference part.

Due to either the requirement of efficiency in solving a problem or the structure of the problem's graphical model representation, it is not always that case that the above inference problem can solved exactly. Thus inference methods can be divided into

- exact inference,
- and approximate inference.

For a limited class of graphs, exact inference such as variable elimination and sum-product algorithm can be used. Some graphs also allow efficient inference after mild graph modification, e.g. junction tree method. However, the above listed inference problems can only be approximately solved in general graphs. The approximate inference family can be further divided into

- Stochastic Approximation (Particle methods),
- Deterministic Approximation (Variational methods).

Stochastic approximation mainly relies on sample instances to answer queries, where a major challenge lies in how to obtain samples efficiently from a target distribution. Gibbs sampling, importance sampling and Markov Chain Monte Carlo are within this family. On the other hand, deterministic approximations refer to the variational methods, such as mean field approximation, loopy belief propagation, expectation propagation etc. *From the perspectives of methodology, we related work in this thesis locates in the family of variational methods under approximate inference category.*

As for learning in probabilistic graphical models, there are two types of learning problems

- Structure learning,
- Parameter learning.

The first case refer to determine the structure of a graphical model from observations, which is usually reduced to the problem of whether there should be an edge between a pair of nodes in the graphical model. The parameter learning is about to determine the parameter of a probabilistic graphical model (or its coupled distribution), with its graphical structure known. Structure learning is out of the scope of this thesis. The term *learning* in thesis means the estimation of the parameters of a distribution. This problem is mainly discussed in Part II, where we would touch the topics about learning in both undirected and directed graphical models.

As for the learning techniques, the learning principles can categorized into

- Maximal likelihood estimation
- Maximal conditional likelihood

- Bayesian estimation
- Maximal ‘Margin‘
- Maximum entropy

in general. We would touch techniques of the first four cases in Part II.

Publications

The following works ware done during the author’s PhD education.

1. Dong Liu and Lars K Rasmussen. Region-based energy nerual network for approximate inference. Under review, 2020.
   Code: https://github.com/FirstHandScientist/renn (Private, publish later)

2. Dong Liu, Minh Thành Vu, Li Zuxing, and Lars K Rasmussen. α belief propagation for approximate beyeisian inference. Under review, 2020.
   Code: https://github.com/FirstHandScientist/alpha-bp

3. Dong Liu, Antoine Honoré, Saikat Chatterjee, and Lars K Rasmussen. Powering hidden markov model by neural network based generative models. In the 24th European Conference on Artificial Intelligence (ECAI), 2020.
   Code: https://github.com/FirstHandScientist/genhmm

4. Dong Liu, Minh Thành Vu, Saikat Chatterjee, and Lars K Rasmussen. Neural network based explicit mixture models and expectation-maximization based learning. In International Joint Conference on Neural Networks, Glasgow, UK, July 2020.
   Code: https://github.com/FirstHandScientist/EM-GM

5. Antoine Honoré, Dong Liu, David Forsberg, Karen Coste, Eric Herlenius, Saikat Chatterjee, and Mikael Skoglund. Hidden markov models for sepsis detection in preterm infants. In 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),2020.

6. D. Liu, N. N. Moghadam, L. K. Rasmussen, J. Huang, and S. Chatterjee. α belief propagation as fully factorized approximation. In 2019 IEEE Global Conference on Signal and Information Processing (GlobalSIP), pages 1-5, 2019.
   Code: https://github.com/FirstHandScientist/amp

7. Dong Liu, Minh Thành Vu, Saikat Chatterjee, and Lars K Rasmussen. Entropy-regularized optimal transport generative models. In 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 3532-3536, 2019.
   Code: https://github.com/FirstHandScientist/eotgms

8. Dong Liu, Baptiste Cavarec, Lars K Rasmussen, and Jing Yue. On dominant interference in random networks and communication reliability. In ICC 2019 IEEE International Conference on Communications (ICC), pages 1-7, 2019.

9. Dong Liu, Chao Wang, and Lars K Rasmussen. Discontinuous reception for multiple-beam communication. IEEE Access, pages 46931-46946, 2019.

10. Dong Liu, Viktoria Fodor, and Lars K Rasmussen. Will scale-free popularity develop scale-free geo-social networks? IEEE Transactions on Network Science and Engineering, 2018.

The material presented in this thesis is based on the author's works which are partially published in 1, 2, 3, 4, 5, 6, 7 listed above.

Outline of Thesis

To be written after main content is finished.

Chapter 2

Background

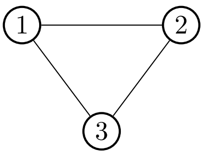
In this chapter, we review some background knowledge that is going be used in this thesis. We begin with the introduction to probabilistic graphical models. Then a divergence measure is introduced. Common inference tasks and methods are discussed before the learning problems in probabilistic graphical models are reviewed, which are interpreted as minimization of the divergence measure.

2.1 Graphical Models

Graphical models provide a formal graph representation of statistical dependency of complex problems or systems. The conditional independence of random variables can be conveniently encoded and analyzed by a graphical model. More importantly, query problems can be resolved by interactions of local regions of a graphical model in exact or approximate ways, which are usually unfeasible to solve directly.

More formally, a graphical model is a graphical representation of a collection of random variables (along their domains) where their statistical dependency encoded into a set of non-negative functions and the graphical structure. Let $\boldsymbol{x} = (x_1, x_2, \cdots, x_N)$ be a vector of random variables with N as a positive integer, where an element variable x_i can be either discrete or continuous random variable and takes values from its domain $\mathcal{X}_i$. Note that the domain of a random variable is not necessary be the same as that of another. With some abuse of notation, we might use $\boldsymbol{x}$ to denote its assignment when there is no cause of ambiguity in context. The joint probability is denoted by $p(\boldsymbol{x}) = p(x_1, x_2, \cdots, x_N)$. We denote $\boldsymbol{\mathcal{X}} = \prod_{x=1}^{N} \mathcal{X}_i$ and then $\boldsymbol{x} \in \boldsymbol{\mathcal{X}}$.

As motivated in Chapter 1.1, a graphical model can be directed or undirected. A directed graphical model also is also know as a Bayesian network or generative model in literature [3, Chapter 8]. We might use the names alternatively. The non-negative functions in graphical models encodes the local compatibility of states of random variables. In directed graphical models, i.e. Bayesian networks, the local functions are conditional probability functions. The joint probability distribution



x_1	x_2	$\varphi(x_1, x_2)$
0	0	10
0	1	1
1	0	1
1	1	10

$\cdots$

x_2	x_3	$\varphi(x_1, x_2)$
0	0	5
0	1	3
1	0	3
1	1	5

Figure 2.1: A Markov random field with three binary nodes. Potential factors are represented by tables.

is represented as the product of these conditional probability functions,

$$p(\boldsymbol{x}) = \prod_{n=1}^{N} p(x_n|\mathcal{P}(x_n)), \tag{2.1}$$

where $\mathcal{P}(\cdot)$ denotes the set of parent nodes in the directed graph. In an directed graphical model, the local functions, i.e. the conditional probability distributions, e.g. $\{p(x_n|\mathcal{P}(x_n))\}$, are normalized and proper distributions. Additionally, sampling from a underlining distribution $p(\boldsymbol{x})$ of a directed graphical model is efficient. Due to acyclic property of directed graphical models, by the well know *ancestral sampling*, a sample $(x_1, x_2, \cdots, x_N)$ can be drawn sequentially via following the directed edges. In another word, x_n is always sampled after $\mathcal{P}(x_n)$. This process might be viewed as the 'generative' process of signal $\boldsymbol{x}$, i.e. how $\boldsymbol{x}$ is generated from the graphical model.

A Bayesian network (generative model) is usually easier to be interpreted due to the fact that its local functions are conditional probabilities and it is natural to decompose the joint underlining distribution into conditional probability distributions. But Bayesian networks can only be applied to the limited cases where influence between variables is directional. In many practical cases, interaction between variables can not be naturally described by impact with directionality. A problem of this kind can be represented by an undirected graphical model, i.e. a Markov random field (MRF). Under certain condition, a Bayesian network can be perfectly represented by a Markov random field without loss of independence information by moralizing edges [20, Chapter 4.5]. Instead of conditional probability distributions, the local functions of MRF represents the compatibility of states of different variables, which are termed as *potential factors*. Different from conditional probabilities in a Bayesian network, a potential factor in a MRF is not necessary normalized (not necessary to be summed to one). We provide a toy example of MRF with three variable nodes as follows.

Example 2. *As shown in Figure 2.1, the MRF encodes dependency of three random variables x_1, x_2, and x_3, where node i is associated with variable x_i and each has a binary domain, i.e. $\mathcal{X}_i = \{0, 1\}$ for $i = 1, 2, 3$. Three potential factors of the MRF* *together define the joint distribution*

$$p(\boldsymbol{x}) = \frac{1}{Z}\varphi_{1,2}(x_1, x_2)\varphi_{2,3}(x_2, x_3)\varphi_{1,3}(x_1, x_3)$$

where $Z = \sum_{x_1,x_2,x_3} \varphi_{1,2}(x_1, x_2)\varphi_{2,3}(x_2, x_3)\varphi_{1,3}(x_1, x_3)$ *normalizes the potential factors such that* $p(\boldsymbol{x})$ *is a proper distribution. The exemplified potential factors in Figure 2.1 demonstrate that it is more compatible or likely when* x_1*,* x_2 *and* x_3 *are in the same state (either* 0 *or* 1*) than they are configured into different states.*

From the above example to a formal statement, a MRF over random vector $\boldsymbol{x}$ can be represented by a undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, with each node $i \in \mathcal{V}$ is associated with a random variable x_i and undirected edge set $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$. This MRF encodes a collection of distributions that factorize as

$$p(\boldsymbol{x}; \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \prod_{\alpha \in \mathcal{I}} \varphi_\alpha(\boldsymbol{x}_\alpha; \boldsymbol{\theta}), \tag{2.2}$$

where $\mathcal{I}$ is the set of indexes of potential factors, and each factor φ_α for $\alpha \in \mathcal{I}$ is defined on subset of $\boldsymbol{x}$, i.e. $\varphi_\alpha : \mathcal{X}_\alpha \to \mathbb{R}^+ \cup \{0\}$, where $\mathcal{X}_\alpha = \prod_{i \in \alpha} \mathcal{X}_i$ is the domain of potential factor φ_α. The scope of factor α is $\boldsymbol{x}_\alpha = \{x_i | i \in \alpha\}$ where $i \in \alpha$ stands for that the variable x_i associated with node i is an argument of potential factor φ_α. In (2.2),

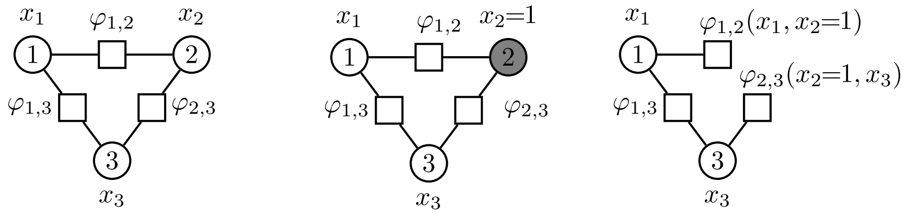
$$Z(\boldsymbol{\theta}) = \sum_{\boldsymbol{x}} \prod_{\alpha \in \mathcal{I}} \varphi_\alpha(\boldsymbol{x}_\alpha; \boldsymbol{\theta}) \tag{2.3}$$

is the *partition function.* Apparently, the partition function normalizes the potential factors such that $p(\boldsymbol{x}; \boldsymbol{\theta})$ is a proper probability.

Remark 1. *We can compare directed and undirected graphical models with regarding to the following aspects.*

- *Representation: The structure and the parameterizaiton in directed graphical models provide a natural representation for many types of real-world domains. MRF representation is not usually as intuitive as that of directed graphical models. But the acyclic property of directed graphical models limits their representation power. On the other hand, MRFs can be either cyclic or acyclic, which offers the flexibility of graph structure and can simplify the graphical representation. Due to the weaker requirement of potential factors and the weaker requirement of graphical structure in MRFs than local functions and acyclic constraint in directed graphical models, respectively, the representation of MRFs are richer.*

- *Local nonnegative functions: The local functions are conditional probability functions in directed graphical models, but potential factors (nonnegative) in undirected cases.*



(a) A factor graph representation.

(b) Conditioning on $x_2 = 1$ in factor graph.

(c) The reduced graph of Figure 2.2b

Figure 2.2: A Markov random field is represented by a factor graph, i.e. 2.2a, conditioning of the MRF 2.2b, the reduced MRF 2.2c.

- *Sampling: Sampling is more straightforward within generative models (directed graphs) than that in MRFs.*

- *Normalization: Since each local function is a conditional probability function in directed graphical models, partition function for normalization is not needed. A MRF in general comes with partition functions, since potential factors are not necessarily normalized.*

Alternative Representation of MRF

The representation of a MRF by $\mathcal{G}(\mathcal{V}, \mathcal{E})$ as explained above is compact, but the potential factors are not present in the graphical representation. An alternative representation to MRF is *factor graph* [21], which is a bipartite graph topology. In a factor graph, a potential factor is explicitly represented as a factor node, as counterpart of variable node associated with a random variable.

Definition 1. *A factor graph $\mathcal{G}_F$, is a bipartite graph that represents the factorization structure of* (2.2)*. A factor graph has two types of nodes: i) a variable node for each variable x_i; ii) a factor node for each potential function φ_α. An edge between a variable node i and factor node α if and only if x_i is argument of φ_α. We would denote a factor graph by $\mathcal{G}_F(\mathcal{V} \cup \mathcal{F}, \mathcal{E}_F)$ with $\mathcal{V}$ as the set of variable nodes, $\mathcal{F}$ as the set of factor nodes, and $\mathcal{E}_F$ the set of undirected edges.*

Example 3. *Let us represent the Example 2.2a by a factor graph, which is shown in Figure 2.2a. Different from the representation by $\mathcal{G}(\mathcal{V}, \mathcal{E})$ in Figure 2.1, factor nodes are explicitly represented by square nodes.*

Conditioning on Observations in MRFs

It is not rare that a graphical model may contain observed variable. The node set of a MRF can be separated into a subset $\mathcal{V}_O$ of nodes, that are associated with observed variable $\boldsymbol{x}_O$, and a subset $\mathcal{V}_U$ of nodes associated with unobserved variable $\boldsymbol{x}_U$. For an evidence is observed,

$$p(\boldsymbol{x}_U|\boldsymbol{x}_O;\boldsymbol{\theta}) = \frac{p(\boldsymbol{x}_U,\boldsymbol{x}_O;\boldsymbol{\theta})}{p(\boldsymbol{x}_O;\boldsymbol{\theta})} = \frac{Z(\boldsymbol{x}_O,\boldsymbol{\theta})}{Z(\boldsymbol{\theta})}, \tag{2.4}$$

where

$$\begin{aligned}\tilde{p}(\boldsymbol{x};\boldsymbol{\theta}) &= \prod_{\alpha\in\mathcal{I}} \varphi_\alpha(\boldsymbol{x}_\alpha;\boldsymbol{\theta}),\\ Z(\boldsymbol{x}_O,\boldsymbol{\theta}) &= \sum_{\boldsymbol{x}_U} \tilde{p}(\boldsymbol{x};\boldsymbol{\theta}),\\ Z(\boldsymbol{\theta}) &= \sum_{\boldsymbol{x}_O}\sum_{\boldsymbol{x}_U} \tilde{p}(\boldsymbol{x};\boldsymbol{\theta}).\end{aligned} \tag{2.5}$$

This means that a condition probability can be computed by partition function and sub-partition functions. Alternatively, when an evidence $\boldsymbol{e}_O$ (an sample instance of $\boldsymbol{x}_O$) is observed, the conditional probability can be written as

$$p(\boldsymbol{x}_U|\boldsymbol{x}_O=\boldsymbol{e}_O;\boldsymbol{\theta}) = \frac{\tilde{p}(\boldsymbol{x}_U,\boldsymbol{x}_O=\boldsymbol{e}_O;\boldsymbol{\theta})}{\sum_{\boldsymbol{x}_U}\tilde{p}(\boldsymbol{x}_U,\boldsymbol{x}_O=\boldsymbol{e}_O;\boldsymbol{\theta})} \propto \tilde{p}(\boldsymbol{x}_U,\boldsymbol{x}_O=\boldsymbol{e}_O;\boldsymbol{\theta}) \tag{2.6}$$

where $\propto$ stands for propositional to. 2.6 shows an interesting phenomenon for MRF including evidence. It can be understood as clamping nodes in $\mathcal{V}_O$ of the MRF to configuration $\boldsymbol{e}_O$, i.e. the domain of $\boldsymbol{x}_O$ becomes a set containing only one instance $\boldsymbol{e}_O$. For instance, an example of conditioning on a variable node for Example 2 is shown in Figure 2.2b.

In addition to the above intuitions, conditioning can also be understood as a process of reducing the graph of a MRF. When a MRF is conditioned on $\boldsymbol{x}_O$, the variables nodes of set $\mathcal{V}_O$ are removed from $\mathcal{G}$, along with their edges. The potential factors with regarding to $\mathcal{V}_U$ are modified accordingly [20, Chapter 4.2.3]. For instance, the graph including evidence node 2 in Figure 2.2b can be further reduced into a Figure 2.2c. Then any inference applicable to a MRF applies to the MRF with nodes clamped as well. A MRF with several nodes clamped to some evidence can be seen either as a manipulation of its domain or the graph itself.

It can be seen that MRF framework is capable to handle conditioning as well. Therefore, in the following part of the thesis, it might or might not have bee based on conditioning observed variables when a MRF is mentioned.

2.2 Divergence

Before we get into more discussion about inference and learning topics, we firstly introduce the concept of *divergence* measures since principles of both learning and inference are closed related with divergence measure. A divergence measure play a fundamental role when we try to use a probability distribution (over discrete or continue variable) q to approximate another probability distribution p. A divergence measure is used to formally quantity how much information is lost when p is represented by q. Denote $\mathbb{P}$ as the space of measures p and q, i.e. $p, q \in \mathbb{P}$.

Definition 2. *Given the space $\mathbb{P}$ of probability distribution for a random variable $\boldsymbol{x}$, a divergence on this space is defined as a function $D(p\|q) : \mathbb{P} \times \mathbb{P} \to \mathbb{R}^+ \cup \{0\}$ such that $D(p\|q) \geqslant 0$ for all $p, q \in \mathbb{P}$ and $D(p\|q) = 0$ if and only if $p = q$.*

Here we introduce the classic *Kullback-Leibler divergence* [23,24], KL divergence for short, which is one of the most widely used divergence measures in machine learning, statistics and information theory.

Definition 3. *The Kullback-Leibler (KL) divergence on $\mathbb{P}$ is defined as a function $KL(\cdot\|\cdot) : \mathbb{P} \times \mathbb{P} \to \mathbb{R}^+ \cup 0$ with the following form*

$$\mathrm{KL}(p\|q) = \sum_{\boldsymbol{x}} p(\boldsymbol{x}) \log \frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}, \tag{2.7}$$

where log *is the natural loggarithm. Note the sum in 2.7 should be replaced by intergral when p and q are probability density functions.*

KL divergence is not symmetric. In another word, there is no equivalence between $\mathrm{KL}(p\|q)$ and $\mathrm{KL}(q\|p)$ in general.

2.3 Inference Tasks

Given a probability distribution $p(\boldsymbol{x})$ as the underline distribution of a graphical model, inference in general can be divided into four kinds of tasks, as brought up in Chapter 1.2. Our work in this thesis would be closely involved with the problems

- computing the likelihood of observed data or unobserved random variable;
- computing the marginals distribution over a particular subset of nodes, i.e. $p(\boldsymbol{x}_A)$ for $A \in \mathcal{V}$. Note that single-node marginal distribution $p(x_i)$ also belongs to this case;
- computing the conditional distribution a subset of nodes given the configuration of another subset of nodes, i.e. $p(\boldsymbol{x}_A|\boldsymbol{x}_B)$ for $A, B \in \mathcal{V}$ and $A \cap B = \emptyset$;

in MRFs. The above tasks are also close related with the inference of partition function

- Computation of $Z(\boldsymbol{\theta}) = \sum_{\boldsymbol{x}} \prod_{\alpha \in \mathcal{I}} \varphi_\alpha(\boldsymbol{x}_\alpha; \boldsymbol{\theta})$, or sub-partition functions.

In the following section, we would introduce the variational methods for the above tasks in high-level.

2.4 Variational inference

In solving inference tasks, one important technique is based on a variational approach. With $p(\boldsymbol{x};\boldsymbol{\theta})$ as the underlining probability distribution of a graphical model, directly inference with $p(\boldsymbol{x};\boldsymbol{\theta})$ is often unfeasible due to the system represented by the graphical model is too large or complex. It can also be the case that even we know the form of $p(\boldsymbol{x};\boldsymbol{\theta})$, the computation in inference tasks can be prohibitive. In variational approaches, a 'trial' probability distribution $b(\boldsymbol{x})$ is introduced to approximate $p(\boldsymbol{x};\boldsymbol{\theta})$. The trial distribution should be intuitively simpler than $p(\boldsymbol{x};\boldsymbol{\theta})$. *Variational free energy* [37] is a quantity used to find such a approximation. The variational free energy is defined by

$$\begin{aligned} F_V(b) &= \mathrm{KL}(b(\boldsymbol{x})||p(\boldsymbol{x};\boldsymbol{\theta})) - \log Z(\boldsymbol{\theta}) \\ &= \sum_{\boldsymbol{x}} b(\boldsymbol{x}) \log \frac{b(\boldsymbol{x})}{p(\boldsymbol{x};\boldsymbol{\theta})} - \log Z(\boldsymbol{\theta}) \\ &= \sum_{\boldsymbol{x}} b(\boldsymbol{x}) \log \frac{b(\boldsymbol{x})}{\tilde{p}(\boldsymbol{x};\boldsymbol{\theta})}, \end{aligned} \tag{2.8}$$

where $\tilde{p}(\boldsymbol{x};\boldsymbol{\theta}) = \prod_{\alpha\in\mathcal{I}} \psi_\alpha(\boldsymbol{x}_\alpha;\boldsymbol{\theta}_\alpha)$. Since $\mathrm{KL}(b(\boldsymbol{x})\|p(\boldsymbol{x};\boldsymbol{\theta}))$ is always non-negative and is zero if and only if $b(\boldsymbol{x}) = p(\boldsymbol{x};\boldsymbol{\theta})$, we have $F_V(b) \geqslant -\log Z(\boldsymbol{\theta})$, with equality when $b(\boldsymbol{x}) = p(\boldsymbol{x};\boldsymbol{\theta})$.

Remark 2. *Note from 2.8, the minimization w.r.t. b of variational free energy is equivalent to the divergence minimization, i.e. $\mathrm{KL}(b(\boldsymbol{x})||p(\boldsymbol{x};\boldsymbol{\theta}))$, since $\log Z(\boldsymbol{\theta})$ does not depend on b. By observing $F_V(b) = \sum_{\boldsymbol{x}} b(\boldsymbol{x}) \log \frac{b(\boldsymbol{x})}{\tilde{p}(\boldsymbol{x};\boldsymbol{\theta})}$, the free energy minimization guides the choice of b such that b is close to an unnormalized measure $\tilde{p}(\boldsymbol{x};\boldsymbol{\theta})$ in its space.*

Another benefit of 2.8 is that we are able to approximate $p(\boldsymbol{x};\boldsymbol{\theta})$ without inference of the true marginal distributions $\{p(\boldsymbol{x}_\alpha;\boldsymbol{\theta}), \alpha \in \mathcal{I}\}$. Since $\log \tilde{p}(\boldsymbol{x};\boldsymbol{\theta})$ can be formulated as sum of log-potential-factors that are all local functions, the computation of $F_V(v)$ can be done by inference of marginals $\{b_\alpha(\boldsymbol{x}_\alpha), \alpha \in \mathcal{I}\}$ of the approximate distribution, which are tractable.

Remark 3. *Discussion: Since we are essentially approximating distribution p by a distribution b, can we minimize $\mathrm{KL}(p(\boldsymbol{x},\boldsymbol{\theta})\|b(\boldsymbol{x}))$ instead?*

It might be feasible by instinct. But a further check would reveal its infeasibility. The $\mathrm{KL}(p(\boldsymbol{x};\boldsymbol{\theta})\|b(\boldsymbol{x}))$ would inevitable requires the marginals of p and therefore requires the exact inference in p, which are what we are trying to avoid. But it does not mean this divergence is useless. As shall be seen in section 2.5, this type of divergence measure is what we need in model learning.

Variational Free Energy and Mean Field

In mean field approach, a fully-factorized approximation is used,

$$b_{MF}(\boldsymbol{x}) = \prod_{i=1}^{N} b_i(x_i). \tag{2.9}$$

Substituting (2.9) into the variational free energy gives

$$F_{MF}(b) = -\sum_{\alpha\in\mathcal{I}}\sum_{\boldsymbol{x}_\alpha} \log\varphi_\alpha(\boldsymbol{x}_a;\boldsymbol{\theta})\prod_{i\in\alpha} b_i(x_i) + \sum_{i\in\mathcal{V}}\sum_{x_i} b_i(x_i)\log b_i(x_i), \tag{2.10}$$

where $i \in \alpha$ stands for the node i's associated x_i is argument of φ_α. Solving the minimization of F_{MF} w.r.t. $b_{MF}(\boldsymbol{x})$ gives the update rule of mean field

$$\log b_i(x_i) \propto \sum_{\alpha\in\mathrm{ne}_i}\sum_{\boldsymbol{x}_\alpha\backslash x_i} \log\varphi_\alpha(\boldsymbol{x}_\alpha;\boldsymbol{\theta}_\alpha)\prod_{j\in\alpha\backslash i} b_j(x_j), \tag{2.11}$$

where $\mathrm{ne}_i = \{\alpha|i\in\alpha, a\in\mathcal{F}\}$, i.e. the potential factors that has x_i as argument.

Bethe Free Energy and (Loopy) Belief Propagation

Different from the mean field approximation, Bethe approximation also includes the non-single-node beliefs $\{b_\alpha(\boldsymbol{x}_\alpha)\}$ apart from the single-node beliefs $\{b_i(x_i)\}$ [51]. In this case, the Bethe free energy is given by

$$F_{Bethe}(b) = \sum_{\alpha}\sum_{\boldsymbol{x}_\alpha} b_\alpha(\boldsymbol{x}_\alpha)\log\frac{b_\alpha(\boldsymbol{x}_\alpha)}{\varphi_\alpha(\boldsymbol{x}_\alpha)} - \sum_{i=1}^{N}(|\mathrm{ne}_i|-1)\sum_{x_i} b_i(x_i)\log b_i(x_i), \tag{2.12}$$

where $|\cdot|$ stands for carnality. Due to the non-single-node beliefs, there are consistency constrains $\sum_{\boldsymbol{x}_\alpha} b_\alpha(\boldsymbol{x}_\alpha) = \sum_{x_i} b_i(x_i) = 1$, $\forall\ i\in\alpha$ to obey. Then, solving the Bethe free energy minimization problem

$$\begin{aligned}
\min_{\{b_\alpha(\boldsymbol{x}_\alpha)\},\{b_i(x_i)\}} \quad & F_{Bethe}(b)\\
\text{s.t.} \quad & \sum_{\boldsymbol{x}_\alpha\backslash x_i} b_\alpha(\boldsymbol{x}_\alpha) = b_i(x_i),\\
& \sum_{\boldsymbol{x}_\alpha} b_\alpha(\boldsymbol{x}_\alpha) = \sum_{x_i} b_i(x_i) = 1,\\
& 0 \leqslant b_i(x_i) \leqslant 1,\\
& i\in\mathcal{V}, a\in\mathcal{F},
\end{aligned} \tag{2.13}$$

where $\mathcal{V}$ and $\mathcal{F}$ are the set of variable nodes and the set of factor nodes in factor graph as defined in Definition 1, gives the (loopy) BP message-passing rule

$$m_{a\to i}(x_i) \propto \sum_{\boldsymbol{x}_\alpha\backslash x_i}\varphi_\alpha(\boldsymbol{x}_\alpha)\prod_{j\in\alpha\backslash i}\prod_{\alpha'\in\mathrm{ne}_j\backslash\alpha} m_{\alpha'\to j}(x_j). \tag{2.14}$$

2.5 Learning principles

We have touched the learning topic in chapter 1, which is to find the 'best' probability distribution $p(\boldsymbol{x};\boldsymbol{\theta})$ in its space $\mathbb{P}$. To make the discussion more concrete, we assume the domain is governed by a underlying distribution p^* that is induced by a (directed or undirected) graphical model, $\mathcal{M}^* = \{\mathcal{K}^*, \boldsymbol{\theta}^*\}$ with $\mathcal{M}^*$ representing its structure and $\boldsymbol{\theta}^*$ representing its parameter. Here we discuss about *model learning* (parameter learning only). For notation simplicity, we use $p^*(\boldsymbol{x})$ to denote this distribution. We are given a dataset $\mathcal{D} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \cdots, \boldsymbol{x}^M\}$. Following the standard assumption, these sample instances are *independent and identically distributed (i.i.d.)*. The task is then to use the information from the dataset to learn a distribution p within its space $\mathbb{P}$, since the governing distribution $p^*(\boldsymbol{x})$ is not known.

The problem of learning a distribution in $\mathbb{P}$ to approximate p^* can be formulated as density estimation. With the concept of KL divergence in section 2.2, learning of p can be formulated as minimizing the KL divergence

$$\begin{aligned}
&\mathrm{KL}(p^*(\boldsymbol{x})\|p(\boldsymbol{x};\boldsymbol{\theta})) \\
=&\mathbb{E}_{\boldsymbol{x}\sim p^*}\left[\log\frac{p^*(\boldsymbol{x})}{p(\boldsymbol{x};\boldsymbol{\theta})}\right] \\
=&-H(p^*) - \mathbb{E}_{\boldsymbol{x}\sim p^*}\left[\log p(\boldsymbol{x};\boldsymbol{\theta})\right],
\end{aligned} \tag{2.15}$$

where $H(p^*)$ is the entropy of p^*. Due to the property of divergence, the KL divergence in 2.15 is zero if and only if $p(\boldsymbol{x};\boldsymbol{\theta}) = p^*(\boldsymbol{x})$. The last line of 2.15 shows that the negative entropy term does not depends on $p(\boldsymbol{x};\boldsymbol{\theta})$. Thus we can just focus on the expectation term $\mathbb{E}_{\boldsymbol{x}\sim p^*}\left[\log p(\boldsymbol{x};\boldsymbol{\theta})\right]$, which is *expected log-likelihood.* Therefore, we can just use the expected log-likelihood to do model learning instead of minimizing the KL divergence.

Note although we can use the expected log-likelihood for model learning task and even model comparison (comparing a trained model with another one), we loss the information of how close a trained model is to p^*. This is due to the omitting of $H(p^*)$, which is not available.

Since it is not possible to know p^* (otherwise we do not need to learn it), the expected log-likelihood is approximated by sample instances of p^*,

$$\mathcal{L}(\mathcal{D};\boldsymbol{\theta}) = \frac{1}{|\mathcal{D}|}\sum_{\boldsymbol{x}\in\mathcal{D}}\log p(\boldsymbol{x};\boldsymbol{\theta}), \tag{2.16}$$

and

$$\mathbb{E}_{\boldsymbol{x}\sim p^*}(\log p(\boldsymbol{x};\boldsymbol{\theta})) \approx \mathcal{L}(\mathcal{D};\boldsymbol{\theta}). \tag{2.17}$$

Log-likelihood $\mathcal{L}(\mathcal{D};\boldsymbol{\theta})$ is one of the most widely used loss for model learning. However, $\mathcal{L}(\mathcal{D};\boldsymbol{\theta})$ is not always a feasible loss to compute due to:

- exact computation of $p(\boldsymbol{x})$ is not possible;

- there are some elements of $\boldsymbol{x}$ which are not observable (latent variables).

For the first case, the typical treatment is to approximate the exact log-likelihood. This is done by approximation with employing inference methods or making simplified assumptions on dependency structure of the graphical model of $p(\boldsymbol{x})$. Then, optimization is carried out with regarding to the approximated log-likelihood. These methods include surrogate likelihood [29, 48], pseudo-likelihood [25, 40], piecewise likelihood [28, 45], saddle-point approximation [44, 50].

Apart from the above case where all variables are observable, the partial observed models,the latent variable case, are equally import in inference and learning with uncertainty. This class of models includes (but not limited to) classic Gaussian mixture models (GMMs) and hidden Markov models (HMMs). There are latent variables because:

- Use of abstract variable to model the generative process (usually a directed graph) of observation data, such as HMMs.

- A practical true attribute of an object may be difficult or impossible to measure exactly. For instance, the disease infection can only be diagnosed via the relevant symptoms, e.g. Example 1; In the position tracking of a car with noisy sensors, the true position of the car might only be inferred via noisy data of sensors.

- No measurement on an attribute of an object of interest is made. For instance, the velocity sensor in the car tracking example might not be stable and might read not quantity now and then.

In general, latent variables are commonly used to deal with partial observation problems, data clustering, data manipulation, etc. Let us denote the observable variable and latent variable by $\boldsymbol{x}_O$ and $\boldsymbol{x}_U$, respectively. We can see that the log-likelihood $p(\boldsymbol{x}_O, \boldsymbol{x}_U; \boldsymbol{\theta})$ is not available any more as it is in the fully-observed case. To deal with the latent variables, we can try to optimize the partial log-likelihood

$$\begin{aligned} l(\boldsymbol{x}_O; \boldsymbol{\theta}) =& \log p(\boldsymbol{x}_O; \boldsymbol{\theta}) \\ =& \mathbb{E}_{q(\boldsymbol{x}_U|\boldsymbol{x}_O)} \left[\log \frac{q(\boldsymbol{x}_U|\boldsymbol{x}_O)}{p(\boldsymbol{x}_U|\boldsymbol{x}_O; \boldsymbol{\theta})} \cdot \frac{p(\boldsymbol{x}_U, \boldsymbol{x}_O; \boldsymbol{\theta})}{q(\boldsymbol{x}_U|\boldsymbol{x}_O)} \right] \\ =& \mathrm{KL}(q(\boldsymbol{x}_U|\boldsymbol{x}_O) \| p(\boldsymbol{x}_U|\boldsymbol{x}_O; \boldsymbol{\theta})) + F(q, \boldsymbol{\theta}) \end{aligned} \tag{2.18}$$

with

$$F(q, \boldsymbol{\theta}) = \mathbb{E}_{q(\boldsymbol{x}_U|\boldsymbol{x}_O)} \left[\log p(\boldsymbol{x}_U, \boldsymbol{x}_O; \boldsymbol{\theta}) \right] + H(q(\boldsymbol{x}_U|\boldsymbol{x}_O)) \tag{2.19}$$

where $H(q(\boldsymbol{x}_U|\boldsymbol{x}_O))$ is the entropy of $q(\boldsymbol{x}_U|\boldsymbol{x}_O)$, and q can be any distribution over $\boldsymbol{x}_U$. Due to the non-negative property of KL divergence, we have

$$l(\boldsymbol{x}_O; \boldsymbol{\theta}) \geqslant F(q, \boldsymbol{\theta}), \tag{2.20}$$

with equality when $q(\boldsymbol{x}_U|\boldsymbol{x}_O) = p(\boldsymbol{x}_U|\boldsymbol{x}_O;\boldsymbol{\theta})$. $F(q,\boldsymbol{\theta})$ is also called *variational lower bound.*

One of the most wide used methods in learning with latent variable is *expectation maximization (EM)* [8]. In EM method, the posterior of $\boldsymbol{x}_U$ is computed exactly from p, $q(\boldsymbol{x}_U|\boldsymbol{x}_O) = \operatorname{argmax}_q F(q,\boldsymbol{\theta}) = p(\boldsymbol{x}_U|\boldsymbol{x}_O;\boldsymbol{\theta})$, which is the optimal solution to q. Then the parameter $\boldsymbol{\theta}$ of p is optimized. The two steps are optimized iteratively.

In cases the posterior of $\boldsymbol{x}_U$ is not feasible to compute, the variational EM [47, section 6.2.2] or Monte Carlo EM (need sampling technique) [35]. There are also neural network based methods with Monte Carlo estimator to cope with the latent variable problems, see [15,19,22,25].

In above discussion, we have assumed that the model's distribution q is explicitly defined, i.e. the distribution (along its density function or mass function) is available.

In an alternative track, the deep generative models grow popularity in literature and can be applied to different areas such as high-dimensional data representation, reinforcement learning and semi-supervised learning, because of its efficient sampling of multi-mode distributions [14]. A generator in a deep generative model induces a distribution that can be either explicit or implicit distribution. The former problem dates back to [7] and receives more attention in recent years with latest work such as varational autoencoder [19], Real-NVP [9], normalizing flow model [18] and neural ordinary difference equations [4]. The training of these models are still based on the maximum likelihood principle or varational likelihood bound.

The later case brings an implicit distribution, where the maximum likelihood principle is not applicable any more. In this case, a state-of-art method is the generative adversarial model that employs a discriminator to play the role of divergence measure [1,15,42], which is essentially explained by a process of the minimization of the Jensen-Shannon divergence. Additionally, other sample-test based distances are employed as alternative methods for implicit model learning. Among this family, optimal transport is receiving more attention in recent years [5, 43] in this track, which has also been applied for training of Boltzmann machine [33] auto-encoders [46] and generative adversarial networks [2].

Part I

Inference

Chapter 3

An alternative view of belief propagation

Content:p

1. α Belief Propagation as Fully Factorized Approximation, GlobalSIP 2019.
2. α Belief Propagation for Approximate Bayesian Inference, under review.

high-level idea of the convergence study, summary of the convergence study here, refer to [20, section 11.3.4]

3.1 Introduction

Bayesian inference provides a general mathematical framework for many learning tasks such as classification, denoising, object detection, and signal detection. The wide applications include but not limited to imaging processing [54], multi-input-multi-output (MIMO) signal detection in digital communication [6, 17], inference on structured lattice [12], machine learning [27, 34, 53]. Generally speaking, a core problem to these applications is the inference on statistical properties of a (hidden) variable $\boldsymbol{x} = (x_1, \ldots, x_N)$ that usually can not be observed. Specifically, practical interests usually include computing the most probable state $\boldsymbol{x}$ given joint probability $p(\boldsymbol{x})$, or marginal probability $p(\boldsymbol{x}_c)$, where $\boldsymbol{x}_c$ is subset of $\boldsymbol{x}$. Naively searching for solutions of these inference problems can be prohibitively expensive in computation.

Probabilistic graphical models defined as structured graphs provide a framework for modeling the complex statistic dependency between random variables. Graphical models are only useful when combined with efficient algorithms. Belief propagation (BP) is a fundamental message-passing algorithm used with graphical models. BP locally exchanges beliefs (statistical information) between nodes [3, 21]. BP can solve inference problems in linear-time exactly when graphs are loop-free or tree-structured [21]. However, many real-world signals are naturally modeled by graph representations with loops. Although BP is still a practical method to do inference approximately (loopy BP) by running it as if there were no loop, its performance varies from case to case and is not guaranteed in general. A direct workaround method to this problem is the general belief propagation [52] or the junction tree algorithm [47], which try to cluster multiple nodes into super nodes to eliminate loops. The difficulty then lies in the graph modification and inference within super nodes when graphs are dense.

Apart from the practical performance issues of BP in loopy graphs, the understanding of it is also limited. [52] shows that BP in loopy graphs approaches to a stationary point of the Bethe free energy approximately. Based on this understanding, variants of BP are derived to improve BP. For instance, fractional BP in [49] applies a correction coefficient to each factor; generalized BP in [52] propagates belief between different regions of a graph; and damping BP in [39] updates beliefs by combining old and new beliefs. Another track falls to the variational method framework, introduced by Opper and Winther [38] and Minka [30,31], namely expectation propagation (EP). In EP, a simpler factorized distribution defined in exponential distribution family is used to approximate the original complex distribution, and an intuitive factor-wise refinement procedure is used to find such an approximate distribution. The method intuitively minimizes a localized Kullback-Leibler (KL) divergence. This is discussed further in [32] and shows unifying view of message passing algorithms. The following work, stochastic EP by [26], explores EP's variant method for applications to large dataset.

Due to the fundamental role of BP for probabilistic inference and related applications, research of seeking insight of BP performance and study on its convergence have been constantly carried out. [**?**] presents the convergence condition of BP in graphs containing a single loop. Work in [**?**] analyzes the Bethe free energy and offers sufficient conditions on uniqueness of BP fixed point. Closely related to our work, [**?**] studies the sufficient conditions for BP convergence to a unique fixed point (as shall be seen in our paper, our convergence analysis is on a message-passing method that generalizes BP). [36] proposes a BP algorithm for high-dimensional discrete space and gives the convergence conditions of it. [**?**] shows that BP can converge to global optima of Bethe energy when BP runs in Ising models that are ferromagnetic (neighboring nodes prefer to be aligned).

There are also works trying to give insight on variant methods of BP. Namely, [**?**,10] studies the convergence condition of Gaussian BP inference over distributed linear Gaussian models. [41] gives the convergence analysis of a reweighted BP algorithm, and offers the necessary and sufficient condition for subclasses of homogeneous graphical models (with identical potentials).

In this work, to gain better understanding of BP in general graphs, we take the path of variational methods to develop an interpretable variant of BP, which we refer to as α-BP. The intuition of α-BP starts with a surrogate distribution $q(\boldsymbol{x})$. $q(\boldsymbol{x})$ is assumed to be fully factorized and each factor of $q(\boldsymbol{x})$ represents a message in the graphical model representation of a target distribution $p(\boldsymbol{x})$. We derive a message-passing rule that is induced by minimizing a localized α-divergence. The merits of α-BP are as follows:

a. α-BP is derived intuitively as localized minimization of α-divergence between original distribution p and surrogate distribution q.

b. α-BP generalizes the standard BP, since the message-passing rule of BP is a special case of α-BP.

c. α-BP can outperform BP significantly in full-connected graphs while still maintaining simplicity of BP for inference.

Apart from the algorithm itself, we give the convergence analysis of α-BP. Sufficient conditions that guarantee the convergence of α-BP to a unique fixed point, are studied and obtained. It turns out that the derived convergence conditions of α-BP depend on both the graph and also the value of α. This result suggests that proper choice of α can help to guarantee the convergence of α-BP. In addition, performance improvement of α-BP over standard BP is demonstrated in a practical application.

3.2 Preliminary

We provide some preliminaries in this section that are needed in this paper. α-divergence is introduced firstly and then a Markov random field is explained.

Divergence Measures

α-divergence, introduced in [32, 55], is a typical way to measure how different two measures characterized by densities p and q are. The definition of α-divergence is as follows,

$$\mathcal{D}_\alpha(p\|q) = \frac{\int_{\boldsymbol{x}} \alpha p(\boldsymbol{x}) + (1-\alpha)q(\boldsymbol{x}) - p(\boldsymbol{x})^\alpha q(\boldsymbol{x})^{1-\alpha} d\boldsymbol{x}}{\alpha(1-\alpha)}, \tag{3.1}$$

where α is the parameter, p and q are not necessary to be normalized.

KL divergence as another way of characterizing difference of measures, is closely related with α-divergence. KL divergence is defined as

$$KL(p\|q) = \int p(\boldsymbol{x}) \log \frac{p(\boldsymbol{x})}{q(\boldsymbol{x})} d\boldsymbol{x} + \int q(\boldsymbol{x}) - p(\boldsymbol{x}) d\boldsymbol{x}, \tag{3.2}$$

where the $\int q(\boldsymbol{x}) - p(\boldsymbol{x}) d\boldsymbol{x}$ is a correction factor to accommodate possibly unnormalized p and q. The KL divergence can be seen as a special case of α-divergence, by observing $\lim_{\alpha\to 1} \mathcal{D}_\alpha(p\|q) = KL(p\|q)$ and $\lim_{\alpha\to 0} \mathcal{D}_\alpha(p\|q) = KL(q\|p)$ (applying L'Hôpital's rule to (3.1)).

Regarding basic properties of divergence measures, both α-divergence and KL divergence are zero when $p = q$, and they are non-negative. Denote KL-projection by

$$\mathrm{proj}[p] = \operatorname*{argmin}_{q\in\mathcal{F}} KL(p\|q), \tag{3.3}$$

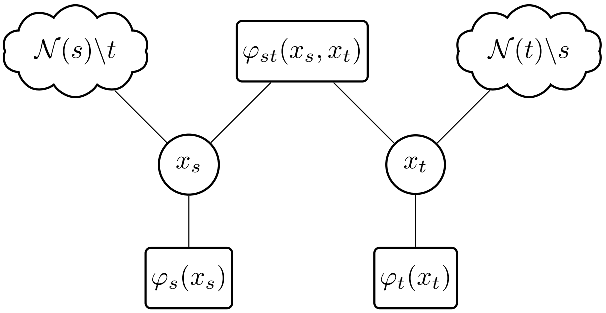


Figure 3.1: Graphic model illustration of $p(\boldsymbol{x})$ in (3.5).

where $\mathcal{F}$ is a family of distribution q. According to the stationary point equivalence Theorem in [32], $\text{proj}[p^\alpha q^{1-\alpha}]$ and $\mathcal{D}_\alpha(p\|q)$ have same stationary points (gradient is zero). A heuristic scheme to find q^* minimizing $\mathcal{D}_\alpha(p\|q)$ starts with an initial q, and repeatedly updates q via the projection on $\mathcal{F}$

$$q(\boldsymbol{x})^{\text{new}} = \text{proj}[p(\boldsymbol{x})^\alpha q(\boldsymbol{x})^{1-\alpha}]. \tag{3.4}$$

This heuristic scheme is a fixed-point iteration. It does not guarantee to converge.

Graphic Models

A graphical model is a probabilistic model that uses a graph to illustrate the statistical dependence between random variables. An undirected graphical model, known as Markov random field (MRF), defines a family of joint probability distributions over random vector $\boldsymbol{x} := (x_1, x_2, \cdots, x_N)$, where each x_i takes values in a discrete finite set $\mathcal{A}$. Let us denote the undirected graph of a MRF by $\mathcal{G} := (\mathcal{V}, \mathcal{E})$. $\mathcal{V} := [1 : N]$ is the node set associated with the index set of entries of $\boldsymbol{x}$. The graph contains undirected edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$, where a pair of $(s, t) \in \mathcal{E}$ if and only if nodes v and u are connected by an edge. In addition to the undirected edge set, let us also define the directed edge set of $\mathcal{G}$ by $\vec{\mathcal{E}}$. We have $|\vec{\mathcal{E}}| = 2|\mathcal{E}|$, where $|\cdot|$ denotes the carnality.

The joint distribution of $\boldsymbol{x}$ can be formulated into a pairwise factorization form as

$$p(\boldsymbol{x}) \propto \prod_{s\in\mathcal{V}} \varphi_s(x_s) \prod_{(s,t)\in\mathcal{E}} \varphi_{st}(x_s, x_t), \tag{3.5}$$

where $\varphi_s : \mathcal{A} \to (0, \infty)$ and $\varphi_{st} : \mathcal{A} \times \mathcal{A} \to (0, \infty)$ are factor potentials. Relation $\propto$ in (3.5) indicates that a normalized factor is needed to turn the right-hand side into a distribution.

The factor graph representation of (3.5) is shown in Figure 3.1. In the figure, $\mathcal{N}(s)$ is the set of variable nodes neighboring x_s via pairwise factors, i.e. $\mathcal{N}(s) = \{t|(t, s) \in \mathcal{E}\}$, and $\backslash$ denotes exclusion.

3.3 α Belief Propagation Algorithm

In this section, we detail the development of the α-BP algorithm. We start with defining a surrogate distribution and then use the surrogate distribution to approximate a given distribution. The message passing rule of α-BP is derived by solving the distribution approximation problem.

Algorithm Development

We begin with defining a distribution

$$q(\boldsymbol{x}) \propto \prod_{s\in\mathcal{V}} \tilde{\varphi}_s(x_s) \prod_{(s,t)\in\mathcal{E}} \tilde{\varphi}_{st}(x_s, x_t), \tag{3.6}$$

that is similarly factorized as the joint distribution $p(\boldsymbol{x})$. The distribution $q(\boldsymbol{x})$ acts as a surrogate distribution of $p(\boldsymbol{x})$. The surrogate distribution would be used to estimate inference problems of $p(\boldsymbol{x})$. We further choose $q(\boldsymbol{x})$ such that it can be fully factorized, which means that $\tilde{\varphi}_{s,t}(x_s, x_t)$ can be factorized into product of two independent functions of x_s, x_t respectively. We denote this factorization as

$$\tilde{\varphi}_{s,t}(x_s, x_t) := m_{st}(x_t) m_{ts}(x_s). \tag{3.7}$$

We use the notation $m_{ts}(x_s)$ to denote the factor as a function of x_s. $m_{ts} : \mathcal{A} \to (0, \infty)$, serves as the message along directed edge $(t \to s)$ in our algorithm. Similarly we have factor or message $m_{st}(x_t)$. Then the marginal can be formulated straightforwardly as

$$q_s(x_s) \propto \tilde{\varphi}_s(x_s) \prod_{w\in\mathcal{N}(s)} m_{ws}(x_s). \tag{3.8}$$

Now, we are going to use the heuristic scheme as in (3.4) to minimize the information loss by using a fully factorized $q(\boldsymbol{x})$ to represent $p(\boldsymbol{x})$. The information loss is measured by α-divergence $\mathcal{D}_\alpha(p(\boldsymbol{x})\|q(\boldsymbol{x}))$.

We perform a factor-wise refinement procedure to update the factors of $q(\boldsymbol{x})$ such that $q(\boldsymbol{x})$ approximates $p(\boldsymbol{x})$. This approach is similar to the factor-wise refinement procedure of assumed density filtering [**?**, **?**] and expectation propagation [30, 32]. Without loss of generality, we begin to refine the factor $\tilde{\varphi}_{ts}(x_t, x_s)$ via α-divergence characterized by α-parameter assigned with α_{ts}. Define $q^{\backslash(t,s)}(\boldsymbol{x})$ as the product of all other factors excluding $\tilde{\varphi}_{ts}(x_t, x_s)$

$$\begin{aligned} q^{\backslash(t,s)}(\boldsymbol{x}) &= q(\boldsymbol{x})/\tilde{\varphi}_{ts}(x_t, x_s) \\ &\propto \prod_{s\in\mathcal{V}} \tilde{\varphi}_s(x_s) \prod_{(v,u)\in\mathcal{E}\backslash(t,s)} \tilde{\varphi}_{vu}(x_v, x_u). \end{aligned} \tag{3.9}$$

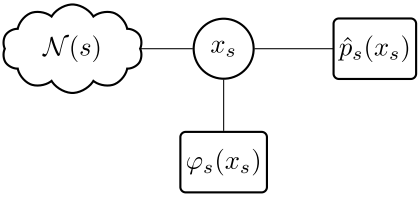


Figure 3.2: Modified graphical model with prior factor.

We also exclude the factor $\varphi_{ts}(x_t, x_s)$ in $p(\boldsymbol{x})$ to obtain $p^{\backslash(t,s)}(\boldsymbol{x})$. Instead of updating $\tilde{\varphi}_{ts}(x_t, x_s)$ directly by solving

$$\operatorname*{argmin}_{\tilde{\varphi}_{ts}^{\text{new}}(x_t,x_s)} \mathcal{D}_{\alpha_{ts}}\Big(p^{\backslash(t,s)}(\boldsymbol{x})\varphi_{ts}(x_t, x_s)\|q^{\backslash(t,s)}(\boldsymbol{x})\tilde{\varphi}_{ts}^{\text{new}}(x_t, x_s)\Big), \tag{3.10}$$

we consider the following tractable problem

$$\operatorname*{argmin}_{\tilde{\varphi}_{ts}^{\text{new}}(x_t,x_s)} \mathcal{D}_{\alpha_{ts}}\Big(q^{\backslash(t,s)}(\boldsymbol{x})\varphi_{ts}(x_t, x_s)\|q^{\backslash(t,s)}(\boldsymbol{x})\tilde{\varphi}_{ts}^{\text{new}}(x_t, x_s)\Big), \tag{3.11}$$

which searches for new factor $\tilde{\varphi}_{ts}^{\text{new}}(x_t, x_s)$ such q can approximate p better. In (3.11), $\mathcal{D}_{\alpha_{ts}}(\cdot)$ denotes the α-divergence with the correcponding parameter α_{ts}. Note that the approximation (3.11) is accurate when $q^{\backslash(t,s)}(\boldsymbol{x})$ is equal to $p^{\backslash(t,s)}(\boldsymbol{x})$. Using fixed-point update in (3.4), the problem in (3.11) is equivalent to

$$\begin{aligned} &q^{\backslash(t,s)}(\boldsymbol{x})\tilde{\varphi}_{ts}^{\text{new}}(x_t, x_s) \propto \\ &\text{proj}\left[q^{\backslash(t,s)}(\boldsymbol{x})\varphi_{ts}(x_t, x_s)^{\alpha_{ts}}\tilde{\varphi}_{ts}(x_t, x_s)^{1-\alpha_{ts}}\right]. \end{aligned} \tag{3.12}$$

Without loss of generality, we update m_{ts} and define

$$\tilde{\varphi}_{ts}^{\text{new}}(x_t, x_s) = m_{ts}^{\text{new}}(x_s)m_{st}(x_t). \tag{3.13}$$

Since KL-projection onto a fully factorized distribution reduces to matching the marginals, substituting (3.13) into (3.12), we obtain

$$\begin{aligned} &\sum_{\boldsymbol{x}\backslash x_s} q^{\backslash(t,s)}(\boldsymbol{x})\tilde{\varphi}_{ts}^{\text{new}}(x_t, x_s) \\ &\propto \sum_{\boldsymbol{x}\backslash x_s} q^{\backslash(t,s)}(\boldsymbol{x})\varphi_{ts}(x_t, x_s)^{\alpha_{ts}}\tilde{\varphi}_{ts}(x_t, x_s)^{1-\alpha_{ts}}. \end{aligned} \tag{3.14}$$

We use summation here. But it should be replaced by integral if $\mathcal{A}$ is a continuous set. Solving (3.14) gives the message passing rule as

$$\begin{aligned} m_{ts}^{\text{new}}(x_s) \propto & m_{ts}(x_s)^{1-\alpha_{ts}}\Bigg[\sum_{x_t}\varphi_{ts}(x_t, x_s)^{\alpha_{ts}}m_{st}(x_t)^{1-\alpha_{ts}} \\ & \tilde{\varphi}_t(x_t)\prod_{w\in\mathcal{N}(t)\backslash s} m_{wt}(x_t)\Bigg]. \end{aligned} \tag{3.15}$$

As for the singleton factor $\tilde{\varphi}_t(x_t)$, we can do the refinement procedure on $\tilde{\varphi}_t(x_t)$ in the same way as we have done on $\tilde{\varphi}_{ts}(x_t, x_s)$. This gives us the update rule of $\tilde{\varphi}_t(x_t)$ as

$$\tilde{\varphi}_t^{\text{new}}(x_t) \propto \varphi_t(x_t)^{\alpha_t} \tilde{\varphi}_t(x_t)^{1-\alpha_t}, \tag{3.16}$$

which is the belief from factor $\varphi_t(x_t)$ to variable x_t. Here α_t is the local assignment of parameter α in α-divergence in refining factor $\tilde{\varphi}_t(x_t)$. Note, if we initialize $\tilde{\varphi}_t(x_t) = \varphi_t(x_t)$, then it remains the same in all iterations, which makes

$$\begin{aligned} m_{ts}^{\text{new}}(x_s) \propto & m_{ts}(x_s)^{1-\alpha_{ts}} \Bigg[\sum_{x_t} \varphi_{ts}(x_t, x_s)^{\alpha_{ts}} m_{st}(x_t)^{1-\alpha_{ts}} \\ & \varphi_t(x_t) \prod_{w \in \mathcal{N}(t) \backslash s} m_{wt}(x_t) \Bigg]. \end{aligned} \tag{3.17}$$

In our notations, a factor potential is undirected, i.e. $\varphi_{ts}(x_t, x_s) = \varphi_{st}(x_s, x_t)$ for all $(t, s) \in \mathcal{E}$. When refining factors with α-BP, each factor potential (corresponding to an edge of $\mathcal{G}$) can be associated with a difference setting of α value. In addition we also have $\alpha_{ts} = \alpha_{st}$.

Remarks on α Belief Propagation

As discussed in Section 3.2, $KL(p\|q)$ is the special case of $\mathcal{D}_\alpha(p\|q)$ when $\alpha \to 1$. When restricting $\alpha_{st} = 1$ for all $(s, t) \in \mathcal{E}$, the message-passing rule in (3.17) becomes

$$m_{ts}^{\text{new}}(x_s) \propto \sum_{x_t} \varphi_{st}(x_s, x_t) \varphi_t(x_t) \prod_{w \in \mathcal{N}(t) \backslash s} m_{wt}(x_t), \tag{3.18}$$

which is exactly the messages of standard BP [3]. From this point of view, we can say α-BP is a generalization of BP.

From the practical perspective of view, α-BP as a meta algorithm can be used with other methods in hybrid way. Inspired by [13] and assembling methods [16], we can modify the graphical model shown in Figure 3.1 by adding an extra factor potential $\hat{p}_s(x_s)$ to each x_s. The extra factor potential $\hat{p}_s(x_s)$ acts as prior information that can be obtained from other methods. In other words, this factor potential stands for our belief from exterior estimation. Then we can run our α-BP on the modified graph. The modified graph is shown in Figure 3.2.

Note although mean field method also uses fully-factorized approximation, it is obtained differently from α-BP and its factorization differs from that of α-BP. In addition, α-BP is different from standard BP with damping technique. The later case uses message update rule that differs from (3.18) slightly by the way of assigning updated message. Also, α-BP differs from the tree-reweighted belief propagation [47] by the way of message update rule and also how algorithm is derived. Please refer to the section **??** in the supplementary for detailed discussion.

3.4 Convergence of α-BP with a Binary State Space

In this section we discuss the convergence of α-BP. Without loss of generality, we consider the case of binary $\mathcal{A}$, i.e. $\mathcal{A} = \{-1, 1\}$. The factor potentials are further detailed as

$$\begin{aligned} \varphi_{st}(x_s, x_t) &= \exp\left\{\theta_{st}(x_s, x_t)\right\}, \\ \varphi_s(x_s) &= \exp\left\{\theta_s(x_s)\right\}. \end{aligned} \tag{3.19}$$

Further assume the symmetric property of potentials

$$\begin{aligned} \theta_{ts}(x_t, x_s) &= -\theta_{ts}(x_t, -x_s) = -\theta_{ts}(-x_t, x_s), \\ \theta_s(x_s) &= -\theta_s(-x_s). \end{aligned} \tag{3.20}$$

For notation simplicity, we use $\theta_{ts} = \theta_{ts}(1,1)$ and $\theta_s = \theta_s(1)$. Denote by $\boldsymbol{\alpha}$ the vector of all local assignments of parameter α, i.e. $\boldsymbol{\alpha} = (\alpha_{ts})_{(t,s)\in\mathcal{E}}$, by $\boldsymbol{\theta}$ the vector of all parameters of potentials, i.e. $\boldsymbol{\theta} = (\theta_{ts})_{(t,s)\in\mathcal{E}}$. Define a matrix $\boldsymbol{M}(\boldsymbol{\alpha}, \boldsymbol{\theta})$ of size $|\vec{\mathcal{E}}| \times |\vec{\mathcal{E}}|$, in which its entries are indexed by directed edges $(t \to s)$, as

$$\begin{aligned} &M_{(t\to s),(u\to v)} \\ &= \begin{cases} |1-\alpha_{ts}|, & u = t, v = s, \\ |1-\alpha_{ts}| \tanh|\alpha_{ts}\theta_{ts}|, & u = s, v = t, \\ \tanh|\alpha_{ts}\theta_{ts}|, & u \in \mathcal{N}(t)\backslash s, v = t, \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{3.21}$$

Theorem 1. *For an arbitrary pairwise Markov random field over binary variables, if the largest singular value of matrix $\boldsymbol{M}(\boldsymbol{\alpha}, \boldsymbol{\theta})$ is less than one, α-BP converges to a fixed point. The associated fixed point is unique.*

Proof. Let us define z_{ts} as the log ratio of belief from node t to node s on two states of $\mathcal{A}$, i.e.

$$z_{ts} = \log\frac{m_{ts}(1)}{m_{ts}(-1)}. \tag{3.22}$$

By combining the local message passing rule in (3.17) with (3.22), we obtain a local update function $F_{ts} : \mathbb{R}^{|\vec{\mathcal{E}}|} \to \mathbb{R}$ that maps $\boldsymbol{z} = (z_{ts})_{(t\to s)\in\mathcal{E}}$ to updated z_{ts}, which can be expressed as

$$F_{ts}(\boldsymbol{z}) = (1-\alpha_{ts})z_{ts} + f_{ts}(\boldsymbol{z}), \tag{3.23}$$

where

$$f_{ts}(\boldsymbol{z}) = \log\frac{\exp\left\{2\alpha_{ts}\theta_{ts} + \Delta_{ts}(\boldsymbol{z})\right\} + 1}{\exp\left\{\Delta_{ts}(\boldsymbol{z})\right\} + \exp\left\{2\alpha_{ts}\theta_{ts}\right\}}, \tag{3.24}$$

with

$$\Delta_{ts}(\boldsymbol{z}) = 2\theta_s + (1-\alpha_{ts})z_{st} + \sum_{w\in\mathcal{N}(u)\backslash t} z_{wt}. \tag{3.25}$$

In the following, we use superscript (n) to denotes the n-th iteration. Since f_{ts} is continuous on $\mathbb{R}^{|\vec{\mathcal{E}}|}$ and differentiable, we have

$$\begin{aligned}
&z_{ts}^{(n+1)} - z_{ts}^{(n)} \\
=&(1-\alpha_{ts})(z_{ts}^{(n)} - z_{ts}^{(n-1)}) + f_{ts}(\boldsymbol{z}^{(n)}) - f_{ts}(\boldsymbol{z}^{(n-1)}) \\
\overset{(a)}{=}&(1-\alpha_{ts})(z_{ts}^{(n)} - z_{ts}^{(n-1)}) + \nabla f_{ts}(\boldsymbol{z}^{\lambda})^T(\boldsymbol{z}^{(n)} - \boldsymbol{z}^{(n-1)}),
\end{aligned} \tag{3.26}$$

where (a) follows by the mean-value theorem, $\boldsymbol{z}^{\lambda} = \lambda \boldsymbol{z}^{(n)} + (1-\lambda)\boldsymbol{z}^{(n-1)}$ for some $\lambda \in (0,1)$, and $\nabla f_{ts}(\boldsymbol{z}^{\lambda})$ denotes the gradient of f_{ts} evaluated at $\boldsymbol{z}^{\lambda}$. In further details, ∇f_{ts} is given by

$$\frac{\partial f_{ts}}{\partial z} = \begin{cases} (1-\alpha_{ts})\frac{\partial f_{ts}}{\partial \Delta_{ts}}, & z = z_{st}, \\ \frac{\partial f_{ts}}{\partial \Delta_{ts}}, & z = z_{wt}, w \in N(t)\backslash s. \\ 0, & \text{otherwise.} \end{cases} \tag{3.27}$$

Our target here is to find the condition to make sequence $\left(z_{ts}^{(n+1)} - z_{ts}^{(n)}\right)$ to converge. To this aim we need to bound the term $\nabla f_{ts}(\boldsymbol{z}^{\lambda})^T(\boldsymbol{z}^{(n)} - \boldsymbol{z}^{(n-1)})$ in (3.26). For this purpose, we need two auxiliary functions $H, G : \mathbb{R}^2 \to \mathbb{R}$ from lemma 4 in [41], which are cited herein for completeness

$$\begin{aligned}
H(\mu;\kappa) &:= \log \frac{exp(\mu+\kappa)+1}{exp(\mu)+exp(\kappa)}, \\
G(\mu;\kappa) &:= \frac{exp(\mu+\kappa)}{exp(\mu+\kappa)+1} - \frac{exp(\mu)}{exp(\mu)+exp(\kappa)} \\
&= \frac{\sinh\kappa}{\cosh\kappa + \cosh\mu},
\end{aligned} \tag{3.28}$$

where it holds that $\frac{\partial H(\mu;\kappa)}{\partial \mu} = G(\mu;\kappa)$. Further, it holds that $|G(\mu;\kappa)| \leqslant |G(0,\kappa)| = \tanh(|\kappa|/2)$. Then we have

$$\begin{aligned}
f_{ts}(z) &= H(\Delta_{ts}(\boldsymbol{z}); 2\alpha_{ts}\theta_{ts}), \\
\frac{\partial f_{ts}}{\partial \Delta_{ts}} &= G(\Delta_{ts}(\boldsymbol{z}); 2\alpha_{ts}\theta_{ts}),
\end{aligned} \tag{3.29}$$

which implies

$$\left|\frac{\partial f_{ts}}{\partial \Delta_{ts}}\right| \leqslant \tanh(\alpha_{ts}\theta_{ts}). \tag{3.30}$$

Combining (3.26), (3.27), (3.29) and (3.30), we have

$$\begin{aligned}
&|z_{ts}^{(n+1)} - z_{ts}^{(n)}| \\
&= |(1-\alpha_{ts})(z_{ts}^{(n)} - z_{ts}^{(n-1)}) + \nabla f_{ts}(\boldsymbol{z}^{\lambda})^T(\boldsymbol{z}^{(n)} - \boldsymbol{z}^{(n-1)})| \\
&\leqslant |(1-\alpha_{ts})(z_{ts}^{(n)} - z_{ts}^{(n-1)})| + |\nabla f_{ts}(\boldsymbol{z}^{\lambda})^T(\boldsymbol{z}^{(n)} - \boldsymbol{z}^{(n-1)})| \\
&= |1-\alpha_{ts}||z_{ts}^{(n)} - z_{ts}^{(n-1)}| + |\nabla f_{ts}(\boldsymbol{z}^{\lambda})|^T|\boldsymbol{z}^{(n)} - \boldsymbol{z}^{(n-1)}| \\
&\overset{(a)}{\leqslant} |1-\alpha_{ts}||z_{ts}^{(n)} - z_{ts}^{(n-1)}| \\
&+ |1-\alpha_{ts}|\tanh(|\alpha_{ts}\theta_{ts}|)|z_{st}^{(n)} - z_{st}^{(n-1)}| \\
&+ \sum_{w\in N(t)\backslash s} \tanh(|\alpha_{ts}\theta_{ts}|)|z_{wt}^{(n)} - z_{wt}^{(n-1)}|, \qquad (3.31)
\end{aligned}$$

where step (a) holds by applying (3.27) and (3.30).

Concatenating all $(t \to s) \in \vec{\mathcal{E}}$ for inequality (3.31) gives

$$|\boldsymbol{z}^{(n+1)} - \boldsymbol{z}^{n}| \leqslant \boldsymbol{M}(\alpha, \boldsymbol{\theta})|\boldsymbol{z}^{(n)} - \boldsymbol{z}^{n-1}|, \qquad (3.32)$$

where $\boldsymbol{M}(\boldsymbol{\alpha}, \boldsymbol{\theta})$ is defined in (3.21), and $\leqslant$ in (3.32) denotes the element-wise inequality. From (3.32), we could further have

$$\|\boldsymbol{z}^{(n+1)} - \boldsymbol{z}^{n}\|_p \leqslant \|\boldsymbol{M}(\alpha, \boldsymbol{\theta})(\boldsymbol{z}^{(n)} - \boldsymbol{z}^{n-1})\|_p, \qquad (3.33)$$

where $1 \leqslant p < \infty$, and $\|\cdot\|_p$ denotes the ℓ^p-norm.

When applying $p = 2$ to (3.33), we have

$$\begin{aligned}
\|\boldsymbol{z}^{(n+1)} - \boldsymbol{z}^{(n)}\|_2 &\leqslant \|\boldsymbol{M}(\boldsymbol{\alpha}, \boldsymbol{\theta})(\boldsymbol{z}^{(n)} - \boldsymbol{z}^{(n-1)})\|_2 \\
&\leqslant \lambda^*(\boldsymbol{M})\|\boldsymbol{z}^{(n)} - \boldsymbol{z}^{(n-1)}\|_2, \qquad (3.34)
\end{aligned}$$

where $\lambda^*(\boldsymbol{M})$ denotes the largest singular value of matrix $\boldsymbol{M}(\boldsymbol{\alpha}, \boldsymbol{\theta})$. If the largest singular value of $\boldsymbol{M}$ is less than 1, the sequence $\left(|\boldsymbol{z}^{(n+1)} - \boldsymbol{z}^{(n)}|\right)$ converges to zero in ℓ^2-norm as $n \to \infty$. Therefore, for $\lambda^*(\boldsymbol{M}) < 1$, ℓ^2-norm $\left(\boldsymbol{z}^{(n)}\right)$ is a Cauchy sequence and must converge.

By concatenating local update function (3.23), we have a global update function $\boldsymbol{F} = (F_{ts})_{(t\to s)\in\vec{\mathcal{E}}}$, which defines a mapping from $\mathbb{R}^{|\vec{\mathcal{E}}|}$ to $\mathbb{R}^{|\vec{\mathcal{E}}|}$. $\boldsymbol{F}$ is a continuous function of $\boldsymbol{z}$, we have

$$\boldsymbol{F}(\lim_{n\to\infty} \boldsymbol{z}^{(n)}) = \lim_{n\to\infty} \boldsymbol{F}(\boldsymbol{z}^{(n)}). \qquad (3.35)$$

Assume that $\left(\boldsymbol{z}^{(n)}\right)$ converges to $\boldsymbol{z}^*$. Then

$$\begin{aligned}
\boldsymbol{F}(\boldsymbol{z}^*) - \boldsymbol{z}^* &= \lim_{n\to\infty} \boldsymbol{F}(\boldsymbol{z}^{(n)}) - \lim_{n\to\infty} \boldsymbol{z}^{(n)} \\
&= \lim_{n\to\infty} (\boldsymbol{z}^{(n+1)} - \boldsymbol{z}^{(n)}) \\
&= 0. \qquad (3.36)
\end{aligned}$$

Thus $\boldsymbol{z}^*$ must be a fixed point.

In what follows we show that the fixed point is unique when $\lambda^*(\boldsymbol{M}) < 1$. Assume that there are two fixed points $\boldsymbol{z}_0^*$ and $\boldsymbol{z}_1^*$ for sequence $\left\{\boldsymbol{z}^{(n)}\right\}$. Then we have

$$\begin{aligned} \boldsymbol{F}(\boldsymbol{z}_0^*) &= \boldsymbol{z}_0^*, \\ \boldsymbol{F}(\boldsymbol{z}_1^*) &= \boldsymbol{z}_1^*. \end{aligned} \tag{3.37}$$

Applying (3.34) gives

$$\|\boldsymbol{F}(\boldsymbol{z}_0^*) - \boldsymbol{F}(\boldsymbol{z}_1^*)\|_2 \leqslant \lambda^*(\boldsymbol{M})\|\boldsymbol{z}_0^* - \boldsymbol{z}_1^*\|_2. \tag{3.38}$$

Substituting (3.37) into (3.38) gives

$$\|\boldsymbol{z}_0^* - \boldsymbol{z}_1^*\|_2 \leqslant \lambda^*(\boldsymbol{M})\|\boldsymbol{z}_0^* - \boldsymbol{z}_1^*\|_2, \tag{3.39}$$

which gives us $\boldsymbol{z}_0^* = \boldsymbol{z}_1^*$ and completes the uniqueness of the fixed point. □

Remark 4. *From Theorem 1 we can see that, the sufficient condition for convergence of α-BP is $\lambda^*(\boldsymbol{M}(\boldsymbol{\alpha}, \boldsymbol{\theta})) < 1$. It is interesting to notice that $\lambda^*(\boldsymbol{M}(\boldsymbol{\alpha}, \boldsymbol{\theta}))$ is a function of $\boldsymbol{\alpha}$ from α-divergence and $\boldsymbol{\theta}$ from joint distribution $p(\boldsymbol{x})$. This means that whether α-BP can converge depends on the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ representing the problem $p(\boldsymbol{x})$ and also the choice of $\boldsymbol{\alpha}$. Therefore, proper choice of $\boldsymbol{\alpha}$ can guarantee the convergence of α-BP if the sufficient condition can possibly be achieved for given $\boldsymbol{\theta}$.*

3.5 Alternative Convergence Conditions for α-BP

Given the fact that α-BP would converge if the condition in Theorem 1 is fulfilled, the largest singular value computation for large-sized graph could be nontrivial. In this section, we give alternative sufficient conditions for the convergence of α-BP.

Corollary 1. *α-BP would converge to a fixed point if the condition*

$$\begin{aligned} \max_{u\to v} & |1-\alpha_{uv}| + |1-\alpha_{vu}| \tanh\left(|\alpha_{vu}\theta_{vu}|\right) \\ & + \sum_{w\in\mathcal{N}(v)\setminus u} \tanh\left(|\alpha_{vw}\theta_{vw}|\right) < 1, \end{aligned} \tag{3.40}$$

is fulfilled or the condition

$$\begin{aligned} \max_{t\to s} & |1-\alpha_{ts}|(1+\tanh(|\alpha_{ts}\theta_{ts}|)) \\ & + (|\mathcal{N}(t)|-1)\tanh(|\alpha_{ts}\theta_{ts}|) < 1. \end{aligned} \tag{3.41}$$

is achieved, where $|\mathcal{N}(t)|$ denotes the carnality of the set $\mathcal{N}(t)$. The associated fixed point is unique.

Proof. Setting $p = 1$ to (3.33), we have

$$\|\boldsymbol{z}^{(n+1)} - \boldsymbol{z}^{n}\|_1 \leqslant \|\boldsymbol{M}(\alpha, \boldsymbol{\theta})(\boldsymbol{z}^{(n)} - \boldsymbol{z}^{n-1})\|_1. \tag{3.42}$$

Furthermore, from (3.33), we also have

$$\|\boldsymbol{z}^{(n+1)} - \boldsymbol{z}^{n}\|_\infty \leqslant \|\boldsymbol{M}(\alpha, \boldsymbol{\theta})(\boldsymbol{z}^{(n)} - \boldsymbol{z}^{n-1})\|_\infty, \tag{3.43}$$

where $\|\cdot\|_\infty$ denotes the ℓ^∞-norm. Then we have

$$\begin{aligned}\|\boldsymbol{z}^{(n+1)} - \boldsymbol{z}^{n}\|_1 &\leqslant \|\boldsymbol{M}\|_1 \|(\boldsymbol{z}^{(n)} - \boldsymbol{z}^{n-1})\|_1,\\ \|\boldsymbol{z}^{(n+1)} - \boldsymbol{z}^{n}\|_\infty &\leqslant \|\boldsymbol{M}\|_\infty \|(\boldsymbol{z}^{(n)} - \boldsymbol{z}^{n-1})\|_\infty,\end{aligned} \tag{3.44}$$

where we omit the parameters of $\boldsymbol{M}$ here for simplicity. We can expand the first multiplicand on the right hand side of (3.44) as follows

$$\begin{aligned}\|\boldsymbol{M}\|_1 =& \max_{u\to v} \sum_{t\to s} M_{(t\to s),(u\to v)}\\ =& \max_{u\to v} |1-\alpha_{uv}| + |1-\alpha_{vu}| \tanh|\alpha_{vu}\theta_{vu}|\\ &+ \sum_{w\mathcal{N}(v)\backslash u} \tanh|\alpha_{vw}\theta_{vw}|,\\ \|\boldsymbol{M}\|_\infty =& \max_{t\to s} \sum_{u\to v} M_{(t\to s),(u\to v)}\\ =& \max_{t\to s} |1-\alpha_{ts}|(1+\tanh|\alpha_{ts}\theta_{ts}|)\\ &+ (|\mathcal{N}(t)|-1)\tanh|\alpha_{ts}\theta_{ts}|.\end{aligned} \tag{3.45}$$

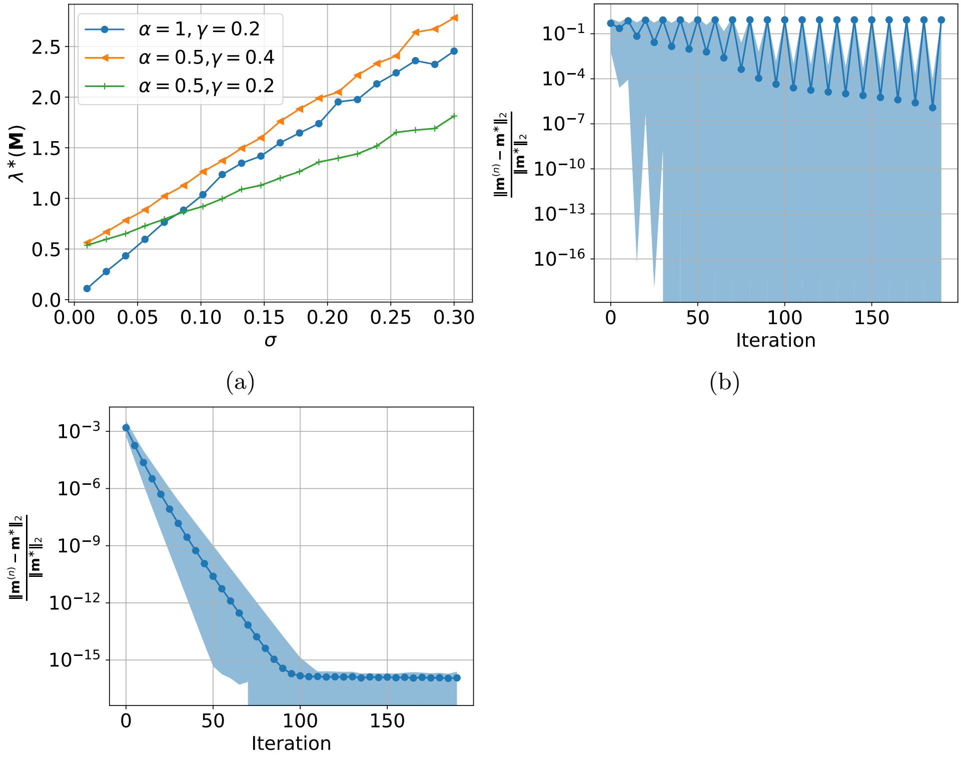
When condition $\|\boldsymbol{M}\|_1 < 1$ is met, sequence $\left(|\boldsymbol{z}^{(n+1)} - \boldsymbol{z}^{n}|\right)$ approaches to zero as $n \to \infty$. Similarly, condition $\|\boldsymbol{M}\|_\infty < 1$ can also guarantee the convergence to zero of sequence $\left(|\boldsymbol{z}^{(n+1)} - \boldsymbol{z}^{n}|\right)$. The analysis for uniqueness of converged fixed point is similar to that in proof of Theorem 1. □

3.6 Experiments

In this section, we firstly give the simulations for convergence condition of α-BP explained in Theorem 1. Then the application of α-BP to a MIMO detection problem is demonstrated. Code is attached along with the submission.

Simulations on Synthetic Problem

In this section simulations on random graphs are carried out to gain some insights on the α-BP. The random graphs used here are generated by Erdos-Rényi (ER)



(a)

(b)

(c)

Figure 3.3: Numerical results on convergence, number of nodes $N = 16$: (**a**) the largest singular value of $\boldsymbol{M}$ defined in (3.21) versus variance of potential parameter $\boldsymbol{\theta}$, normalized error $\|\boldsymbol{m}^{(n)} - \boldsymbol{m}^*\|_2/\|\boldsymbol{m}^*\|_2$ versus the number of iterations. Parameter setting: $\gamma = 0.4$, $\alpha = 1$ (equivalent to standard BP), $\sigma = 0.5$. and (**c**) normalized error $\|\boldsymbol{m}^{(n)} - \boldsymbol{m}^*\|_2/\|\boldsymbol{m}^*\|_2$ versus the number of iterations. Parameter setting: $\gamma = 0.4$, $\alpha = 0.5$, $\sigma = 0.1$. For (b) and (c), blue region denotes the range from minimum to maximum of the normalized error of 100 graph realizations, whereas the curve stands for mean error of the 100 realized graphs.

model [11]. In generating a graph by EP model, an edge between any two nodes is generated with probability γ, $\gamma \in (0, 1)$.

Note that the MRF joint probability in (3.5) can be reformulated into

$$p(\boldsymbol{x}) \propto \exp\{-\boldsymbol{x}^T \boldsymbol{J} \boldsymbol{x} - \boldsymbol{b}^T \boldsymbol{x}\}, \boldsymbol{x} \in \mathcal{A}^N, \tag{3.46}$$

with $\varphi_{ts}(x_t, x_s) = e^{-2J_{t,s}x_t x_s}$ and $\varphi_s(x_s) = e^{-J_{s,s} - b_s x_s}$. J here is the weighted adjacency matrix. In our experiments, we generate a random graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with γ by ER model and then associate potential factors to the graph. Specifically, factor $\varphi_s(x_s)$ is associated to node x_s, $s \in \mathcal{V}$, and φ_{ts} to edge $(t, s) \in \mathcal{E}$. J_{ts} is zero if these is no edge (t, s).

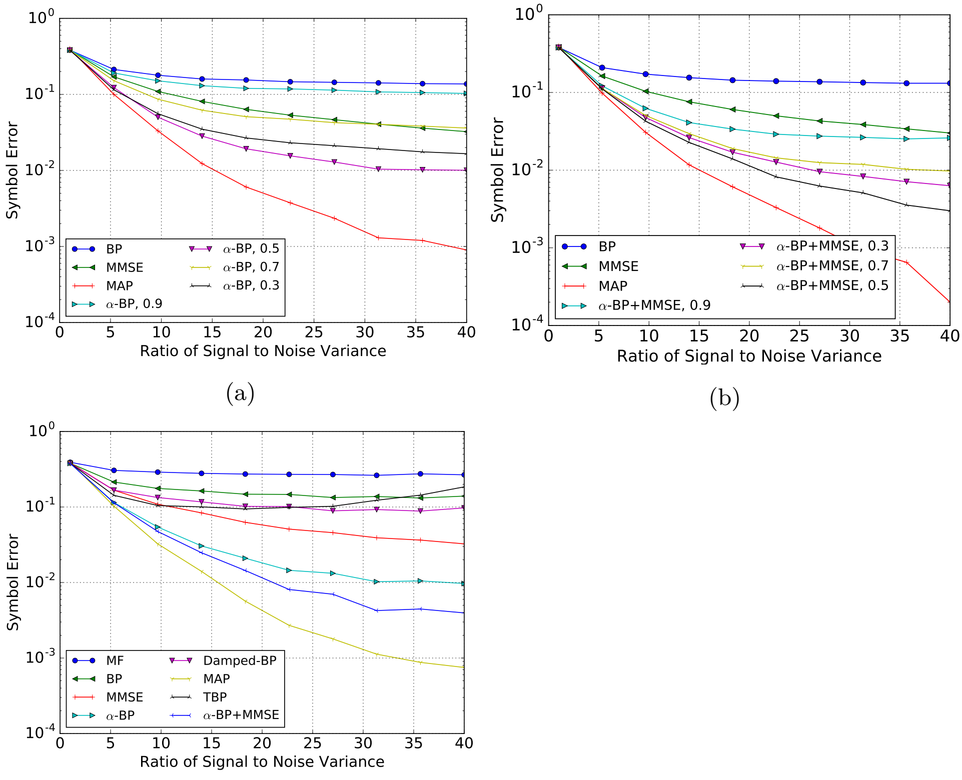
For this set of experiments, we set $\mathcal{A} = \{-1, 1\}$ and $N = 16$. To specify (3.46), the non-zero entries of $\boldsymbol{J}$ is sampled from a Gaussian distribution $\mathsf{N}(0, \sigma^2)$, i.e. $J_{ts} \sim \mathsf{N}(0, \sigma^2)$ if $J_{ts} \neq 0$. For entries of $\boldsymbol{b}$, we use $b_t \in \mathsf{N}(0, (\sigma/4)^2)$. For each edge $(t, s) \in \mathcal{E}$, we set $\alpha_{ts} = \alpha$, i.e. the edges share a global value α.

In Figure 3.3a, we show how the largest singular value of $\boldsymbol{M}(\boldsymbol{\alpha}, \boldsymbol{\theta})$ as defined in (3.21) changes when the standard deviation σ of potential factors increases. The behavior is illustrated with different values of α and the edge probability γ. For each curve, a point on the curve is the mean of 100 realizations of random graphs as described above. The curves of Figure 3.3a show in general that a larger standard deviation of potential factors of graph edges makes it more difficult to fulfill the convergence condition in Theorem 1. This is also the case when a graph is denser as we raise the edge probability γ in generating random graphs, by comparing the green and orange curves. The comparison between green and blue curves indicates that choice of α value in α-BP also makes a difference, and its effect depends on the graph itself. How to tune α value to fulfill condition of Theorem 1 depends not only how dense (γ) the graph is, but also how potential factors spread out from each other.

To illustrate our developed convergence condition for α-BP, we also observe how messages in a graph changes along belief propagation iterations. To be specific, we run our α-BP with 200 iterations on a graph, after which the messages in the graph are denoted by $\boldsymbol{m}^*$. $\boldsymbol{m}^*$ can be the converged messages if α-BP has converged within the 200 iterations. Then we measure the quantity $\|\boldsymbol{m}^{(n)} - \boldsymbol{m}^*\|_2 / \|\boldsymbol{m}^*\|_2$ during the iterations. In Figure 3.3b, we generate 100 random graphs by ER model with parameter setting as $\gamma = 0.4$, $\alpha = 1$[1], $\sigma = 0.5$. By referring to the curves in Figure 3.3a, it can be inferred that this setting does not fulfill the condition in Theorem 1. The log error changes versus iteration number n for the 100 graphs are shown in Figure 3.3b, in which the blue region indicates the range and the solid curve indicates the mean of the normalized errors. It is clear that Figure 3.3b does not show any sign of convergence within 200 iterations.

We then carry out a set of experiments in Figure 3.3c similar to our experiments in Figure 3.3b. The only difference lies in the graph generating process. Here we set the parameters to be $\gamma = 0.4$, $\alpha = 0.5$, $\sigma = 0.1$. According to our curves in Figure 3.3a, a graph generated with this parameter setting should fulfill the condition in Theorem 1. Due to randomness of both graph generating by ER and potential factors, we regenerate a graph if the initial generated graph does not satisfy $\lambda^*(\boldsymbol{M}) < 1$. Therefore the 100 graphs used in experiments for Figure 3.3c all fulfill the Theorem 1. The result in Figure 3.3c is consistent with our analysis on the convergence of α-BP.

[1] $\alpha = 1$ in α-BP corresponding to standard BP.



(a)

(b)

(c)

Figure 3.4: Numerical results of α-BP: symbol error of MIMO detection.

Application to MIMO Detection

In this section, we show the application of α-BP to a MIMO detection problem. For a MIMO system, the observation $\boldsymbol{y}$ is a linear function of channel $\boldsymbol{H} \in \mathbb{R}^{N \times N}$ when unknown signal $\boldsymbol{x}$ need to be estimated,

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{e}, \boldsymbol{x} \in \mathcal{A}^N, \tag{3.47}$$

where $\boldsymbol{e}$ is noise modeled as Gaussian noise $\boldsymbol{e} \sim \mathsf{N}\left(\boldsymbol{0}, \sigma_w^2 \boldsymbol{I}\right)$. Here $\boldsymbol{I}$ is unitary matrix. In this case, the posterior of $\boldsymbol{x}$ can be written as:

$$\begin{aligned} p(\boldsymbol{x}|\boldsymbol{y}) &\propto e^{-\frac{1}{2\sigma_w^2}\|\boldsymbol{H}\boldsymbol{x}-\boldsymbol{y}\|_2^2} \\ &= e^{-\frac{1}{2\sigma_w^2}\left[\boldsymbol{x}^T\boldsymbol{H}^T\boldsymbol{H}\boldsymbol{x}-2\boldsymbol{y}^T\boldsymbol{H}\boldsymbol{x}+\boldsymbol{y}^T\boldsymbol{y}\right]}. \end{aligned} \tag{3.48}$$

Denote $\boldsymbol{S} = \boldsymbol{H}^T\boldsymbol{H}$, $\boldsymbol{h}_i$ as the i-th column of $\boldsymbol{H}$, and

$$\varphi_i(x_i) = e^{-\frac{S_{i,i}x_i^2}{2\sigma_w^2}+\frac{\langle \boldsymbol{h}_i, \boldsymbol{y}\rangle x_i}{\sigma_w^2}}, \varphi_{ij}(x_i, x_j) = e^{-\frac{x_i S_{i,j} x_j}{\sigma_w^2}}. \tag{3.49}$$

Then (3.48) is a realization of (3.5). We set $\mathcal{A} = \{-1, 1\}$, $N = 8$, and $\boldsymbol{H} \in \mathbb{R}^{8\times 8}$ sampled from Gaussian.

We test the application of α-BP to the MIMO signal detection numerically. We run the α-BP, without the prior trick (Subsection 3.3) in Figure 3.4a and with the prior in Figure 3.4b (legend "α-BP+MMSE") as estimation of minimum mean square error (MMSE). The reference results of MMSE and maximum a posterior (MAP, exhausted search) are also reported under the same conditions. MMSE estimator depends on Gaussian posterior $\mathsf{N}(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}})$, where $\hat{\boldsymbol{\mu}} = (\boldsymbol{H}^T\boldsymbol{H} + \sigma_w^2\boldsymbol{I})^{-1}\boldsymbol{H}^T\boldsymbol{y}$ and $\hat{\boldsymbol{\Sigma}} = (\boldsymbol{H}^T\boldsymbol{H} + \sigma_w^2\boldsymbol{I})^{-1}\sigma_w$. Detection of MMSE carried out by $\operatorname{argmin}_{x_i \in \mathcal{A}} |x_i - \hat{\mu}_i|$.

Figure 3.4a shows that BP even underperforms MMSE but α-BP can outperform MMSE by assigning smaller value of α. Note that MMSE requires the matrix inverse computation whose complexity is proportional to N^3, while the complexity of α-BP increases linearly with N. Therefore α-BP is superior to MMSE both performance-wise and complexity-wise. However, there is still a big gap between α-BP (even for $\alpha = 0.5$) and MAP. This gap can be decreased further by using the prior trick discussed in Subsection 3.3. Figure 3.4b exemplifies this effects by using prior belief from MMSE, $\hat{p}_i(x_i) \propto \exp\{-(x_i - \hat{\mu}_i)^2/(2\hat{\boldsymbol{\Sigma}}_{i,i})\}$, by modifying the graph as shown in Figure 3.2, which comes with legend "α-BP+MMSE". It is shown that larger performance gain is observed when α-BP runs with prior belief.

Additional, we also carry out the experiments where the proposed α-BP is compared with mean field (legend 'MF'), BP with damping technique [39] (with legend 'Damped-BP'), and Tree-reweighted belief propagation [47] (with legend 'TBP') in Figure 3.4c. As expected, mean field method performs no better than BP. Damping technique improves BP's performance with a noticeable difference but still falls behind MMSE. The performance of tree-reweighted BP reaches that of MMSE in low ratio range of signal-to-noise variance but degenerates a lot in the high ratio range. The old message and potential factors are reweighted by the edge appearance probability in TBP to compute new messages. In TBP, the edge appearance probability is the probability that the edge exists in a randomly chosen spanning tree from all possible spanning trees of graph $\mathcal{G}$, which is usually expensive to compute.

3.7 Discussion

In this paper, we have developed and analyzed a new alternative to standard BP message passing algorithm. The developed α-BP algorithm has the clear intuition of minimization of a localized α-divergence. Convergence conditions of α-BP are offered in binary state space of each random variable. α-BP is a valid and practical algorithm accoinding to our experiments. With prior trick, the performance of α-BP can be further improved. Future works would be to investigate more general conditions for convergence of α-BP.

Chapter 4

Region-based Energy Neural Network Model

work in Region-based Energy Neural Network for Approximate Inference, under, review

One of the motivations to use region graph: refer to [20, section 11.3.5.3] and figure 11.7 in this section.

4.1 Region-based graph and energy

4.2 RENN model for Approximate Inference

4.3 RENN model for markov random field training

4.4 Experimental results

4.5 Summary

Part II

Learning

Chapter 5

Learning with inference

5.1 learning Undirected graphical models/ MRF

move the MRF learning by using RENN here

I should read lecture note 7 of 10-708 again when writing this seciton.

5.2 Amortized/Neural Variational Learning and Inference of partial observed MRF

1. TRW as upper bound to partition function

2. Mean field or negative TRW as lower bound to partition function

combining above together, we can obtain two different lower bound of likelihood. Consider if worthy a paper.

- The log-likelihood of partial observed MRF is non-convex in general (log-sum-exp is convex, but the difference of two log-sum-exp functions might not be). This combination convert the original non-convex learning into convex optimization with regarding to MRF parameter? should be, but need a confirmation.
- 1. The speed of training can be improved by directly optimizing amortized beliefs.
- The bound becomes tighter by using clamping of variable, clamping can be done with or without selection of variables. No sampling is needed in training or inference.
- If need more contribution, use tree-reweighted hyper graph to obtain tighter bound.
- Not necessarily done here: the bound can also be further improved by important sampling.

Reference:

- 1. Wainwright, 2003, Tree-reweighted belief propagation algorithms and approximate ML estimation by pseudo-moment matching
- 2. Weller, 2015,Clamping Improves TRW and Mean Field Approximations
- 3. Mnih, 2014, Neural Variational Inference and Learning in Belief Networks, which describes a neural variational method for belief network. The major difference is the belief network as a DAG do not have the problem of partition function difficulty as MRF or partial observed MRF.

5.3 Notation

Random variable $\boldsymbol{v} \in \mathcal{X}_v$ that can be observed. Random variable $\boldsymbol{h} \in \mathcal{X}_h$ that is hidden variable and can not be observed.

An alternative plan:

- Training RENN with marginal-likelihood instead of joint likelihood, ref to Domke13
- If the above works, use Gaussian kernals to define potential, Marvin T. T. Teichmann Convolutional CRFs for Semantic Segmentation

5.4 Model and Problem Definition

We define the conditional probabilistic model as

$$p(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \tilde{p}(\boldsymbol{v}, \boldsymbol{h} | \boldsymbol{\theta}), \tag{5.1}$$

with

$$Z(\boldsymbol{\theta}) = \sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}} \tilde{p}(\boldsymbol{v}, \boldsymbol{h} | \boldsymbol{\theta}) \tag{5.2}$$

$$\tilde{p}(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta}) = \exp\left\{-E(\boldsymbol{v}, \boldsymbol{h}, \boldsymbol{\theta})\right\} \tag{5.3}$$

where $E(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta})$ is the average energy: $\mathcal{X}_v \times \mathcal{X}_h \to \mathbb{R}$.

We want to maximize the marginal likelihood:

$$\max_{\boldsymbol{\theta}} \log \sum_{\boldsymbol{h}} p(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta}) = \max_{\boldsymbol{\theta}} \log\ Z(\boldsymbol{v}, \boldsymbol{\theta}) - \log\ Z(\boldsymbol{\theta}), \tag{5.4}$$

where $Z(\boldsymbol{v}, \boldsymbol{\theta}) = \sum_{\boldsymbol{h}} \tilde{p}(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta})$

5.5 A lower bound of the marginal likelihood

Denote $A(\boldsymbol{\theta}) = \log Z(\boldsymbol{\theta})$ and $A(\boldsymbol{v}, \boldsymbol{\theta}) = \log Z(\boldsymbol{v}, \boldsymbol{\theta})$

$$E(\boldsymbol{v}, \boldsymbol{h}, \boldsymbol{\theta}) = -\langle \boldsymbol{\theta}, \boldsymbol{\varphi}(\boldsymbol{v}, \boldsymbol{h}) \rangle \tag{5.5}$$

and

$$\boldsymbol{\mu} = \mathbb{E}_{p(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta})}[\boldsymbol{\varphi}(\boldsymbol{v}, \boldsymbol{h})]. \tag{5.6}$$

In case of overcomplete representation of $\boldsymbol{\varphi}$, $\boldsymbol{\mu}$ is the set of marginal distributions.

With mean field approximation,

$$A_M(\boldsymbol{v}, \boldsymbol{\theta}) = \max_{\boldsymbol{\mu}_{\boldsymbol{v}} \in \mathcal{M}_M} \langle \boldsymbol{\theta}, \boldsymbol{\mu}_{\boldsymbol{v}} \rangle + H_M(\boldsymbol{\mu}_{\boldsymbol{v}}), \tag{5.7}$$

where $\mathcal{M}_M$ is the subspace of distributions where each variable is independent. And we have

$$A_M(\boldsymbol{v}, \boldsymbol{\theta}) \leqslant A(\boldsymbol{v}, \boldsymbol{\theta}). \tag{5.8}$$

With tree-reweighted approximation, TRW,

$$A_T(\boldsymbol{\theta}) = \max_{\boldsymbol{\mu} \in \mathcal{M}_T} \langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle + H(\boldsymbol{\mu}), \tag{5.9}$$

where $\mathcal{M}$ is the subspace of distributions where each variable is independent. And we have

$$A_T(\boldsymbol{\theta}) \geqslant A(\boldsymbol{\theta}). \tag{5.10}$$

We define the lower bound of marginal loglikelihood:

$$\mathcal{L}(\boldsymbol{\theta}) = A_M(\boldsymbol{v}, \boldsymbol{\theta}) - A_T(\boldsymbol{\theta}) \leqslant \log \sum_{\boldsymbol{h}} p(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta}). \tag{5.11}$$

Connection to RBM:

$$p(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \exp\{\boldsymbol{v}\boldsymbol{W}\boldsymbol{h} + \boldsymbol{v}\boldsymbol{b} + \boldsymbol{v}\boldsymbol{a}\} \tag{5.12}$$

Note $p(\boldsymbol{h}|\boldsymbol{h})$ is exactly independent, and thus the $A_M(\theta) = A(\boldsymbol{\theta})$ can be achieved, then how tight the lower bound $\mathcal{L}(\boldsymbol{theta})$ would dependent only on the TRW bound.

I should also consider how to use the trained model for prediction.

This is also closely connected to variational see Section 6.2, Wainwright, Graphical Models, Exponential Families, and Variational Inference.

5.6 Experiment

- Start with standard RBM section 4.2 in Amortized learning of MRFs
  - Try to break the conditinal independence by connecting nodes of $\boldsymbol{h}$
  - Extent to conditional RBM training for denoising and data completion
- high-order HMMs

Chapter 6

Powering the expectation maximization method by neural networks

content: Neural Network based Explicit Mixture Models and Expectation-maximization based Learning, under review

section/chapter transition text: mixture model could be obtained from clamping and condition on a discrete variable, ref to Geier, Locally conditional belief propagation. Weller, clamping variables and approximate inference

Remark 5. *Theory interpretation of EM and variational EM, see Section 6.2, Wainwright, Graphical Models, Exponential Families, and Variational Inference.*

Remark 6. *RELATIONSHIP TO K-MEANS CLUSTERING Big picture: The EM algorithm for mixtures of Gaussians is like a soft version of the K-means algorithm.*

Remark 7. *EM lower bound + entropy of posterior of latent variable if a free energy. ref to 10-708 lecture6 note. EM using posterior of latent variable is equivalent to fully observable MLE where statistics are replaced by their expectations w.r.t the posterior.*

Can be viewed as two-node graphical model learning. 10-708lecture5-note

6.1 Normalizing flow

6.2 expectation maximization of neural network based mixture models

6.3 An alternative construction method

6.4 Experiments

6.5 Summary

Chapter 7

Powering Hidden Markov Model by Neural Network based Generative Models

For permeable part and notation of this chapter, refer to [20, Chapter 6.2]. Give a figure/illustration: Dynamic Bayesian Network > 2-TBN > HMM

A bit history of HMM, see [20, Chapter 6.8]

content:

1. Powering Hidden Markov Model by Neural Network based Generative Models, ECAI 2020

2. Antoine Honore, Dong Liu, Hidden Markov Models for sepsis detection in preterm infants, ICASSP, 2020

HMM is an instance of 2-time-slice Bayesian network(2-TBN) (section 6.2.2 Koller). Also, it can be argued from CRF.

7.1 Hidden Markov Model

7.2 GenHMM

7.3 Application to phone recognition

7.4 Application to sepsis detection in preterm infants

7.5 Summary

Chapter 8

An implicit probabilistic generative model

content: Entropy-regularized Optimal Transport Generative Models, ICASSP 2019

8.1 Modeling data without explicit probabilistic distribution

8.2 Employing EOT for modeling

8.3 Experimental results

8.4 Summary

Part III

Epilogue

Chapter 9

Conclusion and Discussions

Bibliography

[1] M. Arjovsky and L. Bottou. Towards Principled Methods for Training Generative Adversarial Networks. *ArXiv e-prints*, January 2017.

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223, 2017.

[3] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

[4] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 6571–6583. Curran Associates, Inc., 2018. URL `http://papers.nips.cc/paper/7892-neural-ordinary-differential-equations.pdf`.

[5] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2292–2300. Curran Associates, Inc., 2013.

[6] J. CÃ©spedes, P. M. Olmos, M. SÃ¡nchez-FernÃ¡ndez, and F. Perez-Cruz. Expectation propagation detection for high-order high-dimensional mimo systems. *IEEE Transactions on Communications*, 62(8):2840–2849, Aug 2014. ISSN 0090-6778.

[7] Gustavo Deco and Wilfried Brauer. Higher order statistical decorrelation without information loss. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 247–254. MIT Press, 1995. URL `http://papers.nips.cc/paper/901-higher-order-statistical-decorrelation-without-information-loss.pdf`.

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B*, 39:1–38, 1977. URL `http://web.mit.edu/6.435/www/Dempster77.pdf`.

[9] L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using Real NVP. *ArXiv e-prints*, May 2016.

[10] Jian Du, Shaodan Ma, Yik-Chung Wu, Soummya Kar, and José M. F. Moura. Convergence analysis of distributed inference with vector-valued gaussian belief propagation. *J. Mach. Learn. Res.*, 18(1):6302–6339, January 2017. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=3122009.3242029`.

[11] Paul Erdos and Alfréd Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60, 1960.

[12] N. Friel, A. N. Pettitt, R. Reeves, and E. Wit. Bayesian inference in hidden markov random fields for binary data defined on large lattices. *Journal of Computational and Graphical Statistics*, 18(2):243–261, 2009. ISSN 10618600. URL `http://www.jstor.org/stable/25651244`.

[13] J. Goldberger and A. Leshem. Pseudo prior belief propagation for densely connected discrete graphs. In *2010 IEEE Information Theory Workshop on Information Theory (ITW 2010, Cairo)*, pages 1–5, Jan 2010.

[14] I. Goodfellow. NIPS 2016 Tutorial: Generative Adversarial Networks. *ArXiv e-prints*, December 2017.

[15] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014. URL `http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf`.

[16] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated, 2014. ISBN 1461471370, 9781461471370.

[17] C. Jeon, R. Ghods, A. Maleki, and C. Studer. Optimality of large mimo detection via approximate message passing. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 1227–1231, June 2015.

[18] D. P. Kingma and P. Dhariwal. Glow: Generative Flow with Invertible 1x1 Convolutions. *ArXiv e-prints*, July 2018.

[19] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[20] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning.* The MIT Press, 2009. ISBN 0262013193.

[21] F. R. Kschischang, B. J. Frey, and H. . Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, Feb 2001. ISSN 0018-9448.

[22] Volodymyr Kuleshov and Stefano Ermon. Neural variational inference and learning in undirected graphical models. *CoRR*, abs/1711.02679, 2017. URL `http://arxiv.org/abs/1711.02679`.

[23] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statist.*, 22(1):79–86, 03 1951. URL `https://doi.org/10.1214/aoms/1177729694`.

[24] Solomon Kullback. *Information Theory and Statistics.* Wiley, New York, 1959.

[25] Miguel Lazaro-Gredilla, Wolfgang Lehrach, and Dileep George. Learning undirected models via query training, 2019.

[26] Yingzhen Li, José Miguel Hernández-Lobato, and Richard E Turner. Stochastic expectation propagation. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2323–2331. Curran Associates, Inc., 2015. URL `http://papers.nips.cc/paper/5760-stochastic-expectation-propagation.pdf`.

[27] Guosheng Lin, Chunhua Shen, Ian Reid, and Anton van den Hengel. Deeply learning the messages in message passing inference. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pages 361–369, Cambridge, MA, USA, 2015. MIT Press. URL `http://dl.acm.org/citation.cfm?id=2969239.2969280`.

[28] Guosheng Lin, Chunhua Shen, Anton van den Hengel, and Ian Reid. Efficient piecewise training of deep structured models for semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[29] You Lu, Zhiyuan Liu, and Bert Huang. Block belief propagation for parameter learning in markov random fields. *CoRR*, abs/1811.04064, 2018. URL `http://arxiv.org/abs/1811.04064`.

[30] Thomas P. Minka. Expectation propagation for approximate bayesian inference. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, UAI '01, pages 362–369, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-800-1. URL `http://dl.acm.org/citation.cfm?id=647235.720257`.

[31] Thomas P. Minka. *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, Cambridge, MA, USA, 2001. AAI0803033.

[32] Tom Minka. Divergence measures and message passing. Technical Report MSR-TR-2005-173, January 2005. URL `https://www.microsoft.com/en-us/research/publication/divergence-measures-and-message-passing/`.

[33] Grégoire Montavon, Klaus-Robert Müller, and Marco Cuturi. Wasserstein training of restricted boltzmann machines. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3718–3726. Curran Associates, Inc., 2016.

[34] G. Montufar. Restricted Boltzmann Machines: Introduction and Review. *ArXiv e-prints*, June 2018.

[35] Ronald C. Neath. On convergence properties of the monte carlo em algorithm, 2012.

[36] N. Noorshams and M. J. Wainwright. Stochastic belief propagation: A low-complexity alternative to the sum-product algorithm. *IEEE Transactions on Information Theory*, 59(4):1981–2000, April 2013.

[37] M. Opper and D. Saad. *Advanced Mean Field Methods: Theory and Practice*. Neural information processing series. MIT Press, 2001. ISBN 9780262150545.

[38] Manfred Opper and Ole Winther. Gaussian processes for classification: Mean-field algorithms. *Neural Comput.*, 12(11):2655–2684, November 2000. ISSN 0899-7667. URL `http://dx.doi.org/10.1162/089976600300014881`.

[39] Marco Pretti. A message-passing algorithm with damping. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(11):P11008–P11008, nov 2005.

[40] Meng Qu, Yoshua Bengio, and Jian Tang. Gmnn: Graph markov neural networks. In *International Conference on Machine Learning*, pages 5241–5250, 2019.

[41] T. G. Roosta, M. J. Wainwright, and S. S. Sastry. Convergence analysis of reweighted sum-product algorithms. *IEEE Transactions on Signal Processing*, 56(9):4293–4305, Sep. 2008. ISSN 1053-587X.

[42] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, *et al.* Improved techniques for training gans. In *Advances in Neural Information Processing Systems 29*, pages 2234–2242. Curran Associates, Inc., 2016.

[43] F. Santambrogio. *Optimal Transport for Applied Mathematicians: Calculus of Variations, PDEs, and Modeling.* Progress in Nonlinear Differential Equations and Their Applications. Springer International Publishing, 2015. ISBN 9783319208282. URL `https://books.google.se/books?id=UOHHCgAAQBAJ`.

[44] Vivek Srikumar, Gourab Kundu, and Dan Roth. On amortizing inference cost for structured prediction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1114–1124, Jeju Island, Korea, July 2012. Association for Computational Linguistics.

[45] Charles A. Sutton and Andrew McCallum. Piecewise training for undirected models. *CoRR*, abs/1207.1409, 2012. URL `http://arxiv.org/abs/1207.1409`.

[46] I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schoelkopf. Wasserstein Auto-Encoders. *ArXiv e-prints*, November 2017.

[47] M. J. Wainwright and M. I. Jordan. *Graphical Models, Exponential Families, and Variational Inference.* now, 2008. URL `https://ieeexplore.ieee.org/document/8187302`.

[48] Martin J. Wainwright. Estimating the "wrong" graphical model: Benefits in the computation-limited setting. *J. Mach. Learn. Res.*, 7:1829–1859, 2006. URL `http://jmlr.org/papers/v7/wainwright06a.html`.

[49] Wim Wiegerinck and Tom Heskes. Fractional belief propagation. In *Proceedings of the 15th International Conference on Neural Information Processing Systems*, NIPS'02, pages 438–445, Cambridge, MA, USA, 2002. MIT Press. URL `http://dl.acm.org/citation.cfm?id=2968618.2968673`.

[50] Sam Wiseman and Yoon Kim. Amortized bethe free energy minimization for learning mrfs. In *Advances in Neural Information Processing Systems 32*, pages 15520–15531. Curran Associates, Inc., 2019.

[51] Jonathan Yedidia, William Freeman, and Yair Weiss. *Understanding belief propagation and its generalizations*, volume 8, pages 239–269. 01 2003. ISBN 1558608117.

[52] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Generalized belief propagation. In *Proceedings of the 13th International Conference on Neural Information Processing Systems*, NIPS'00, pages 668–674, Cambridge, MA, USA, 2000. MIT Press.

[53] KiJung Yoon, Renjie Liao, Yuwen Xiong, Lisa Zhang, Ethan Fetaya, Raquel Urtasun, Richard S. Zemel, and Xaq Pitkow. Inference in probabilistic graphical models by graph neural networks. *CoRR*, abs/1803.07710, 2018.

[54] R. Zhang, C. A. Bouman, J. Thibault, and K. D. Sauer. Gaussian mixture markov random field for image denoising and reconstruction. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 1089–1092, Dec 2013.

[55] Huaiyu Zhu and Richard Rohwer. Information geometric measurements of generalisation. Technical report, 1995.