The Dissertation Committee for Mingzhang Yin
certifies that this is the approved version of the following dissertation:

Variational Methods with Dependence Structure

Committee:

Mingyuan Zhou, Supervisor

Purnamrita Sarkar

Qixing Huang

Peter Mueller

Variational Methods with Dependence Structure

by

Mingzhang Yin

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2020

Dedicated to my loved ones.

Acknowledgments

I always think the pursuit of truth is a path of beauty. I am fortunate to take such a journey at the University of Texas at Austin with many insightful and supportive people, who make the experience memorable.

First and foremost, I want to express my sincere gratitude to my advisor, Dr. Mingyuan Zhou. Back in 2016, I stepped into the deep learning realm and was very new to the independent research. I will never forget the patience he had when I was in difficulties, the firm confidence he gave when I was hesitating and the moments we cheered together for new discoveries. With comprehensive knowledge, he taught me a taste of choosing right problems and an attitude towards science - honest, rigorous and innovative. Also, many exciting projects cannot be completed without his generous allowance and encouragement to explore new areas on my own interest. I appreciate his fostering and protection of my curiosity. In retrospect, I am blessed to have Dr. Zhou as my advisor, to have the opportunity to learn and work with him.

I am also grateful to Dr. Purnamrita Sarkar, for the pure joy I felt when we played math on the whiteboard and for the comfort she gave when I hit rock bottom. I want to thank Dr. George Tucker for hosting my internship at Google Brain where we worked with Doctors Chelsea Finn and Sergey Levine. I learned a lot from the inspiring discussions over hundreds of emails. Thanks Dr.

Corwin Zigler for guiding me into the causal inference in the final semester. I want to thank all my other co-authors: Dr. Bowei Yan, who gave generous help when I was a junior in Ph.D. program; Yuguang Yue, who has been studying with me since the undergraduate and Dr. Y.X. Rachael Wang who gave me detailed mentorship during our collaboration. Thanks to Doctors Stephen G Walker, James Scott, Constantine Caramanis, Peter Müller, Qixing Huang for the wonderful courses and for being in my committee.

I want to take this chance to thank my friends at Austin with whom I spent happy leisure time: Carlos Tadeu Pagani Zanini, Su Chen, Yang Ni, Kelly Kang, Matteo Vestrucci, Vera Liu, Xinjie Fan, Yanxin Li, Zewen Hanxi, Qiaohui Lin, Yan Long, Xiaoyu Qian, Yibo Hu. Thanks to all my friends that are not at Austin; I can feel your support even remotely. I want to thank the warm staffs and the Department of Statistics and Data Science for the bright office and financial supports. Thanks to all the cafes, bookstores and parks in Austin for hosting my pondering upon life's smallest and biggest questions.

My mother, Ying Hu, and father, Cunzhen Yin, you are my role models. I hope my work can be as solid as the water turbines my mother designed, and as useful as the food machines my father guided to construct. Thanks for your everlasting life guidance, incredible support on my study and your caring about my health. This space is too small to contain all my gratitude.

Finally, a very special thank to Jing Zhang. Though often far away, I never felt a second you were not standing by me. You are always in my heart. This thesis is dedicated to you and my parents.

Variational Methods with Dependence Structure

Publication No. _____

Mingzhang Yin, Ph.D.
The University of Texas at Austin, 2020

Supervisor: Mingyuan Zhou

It is a common practice among humans to deduce, to explain and to make predictions based on concepts that are not directly observable. In Bayesian statistics, the underlying propositions of the unobserved latent variables are summarized in the posterior distribution. With the increasing complexity of real-world data and statistical models, fast and accurate inference for the posterior becomes essential. Variational methods, by casting the posterior inference problem in the optimization framework, are widely used for their flexibility and computational efficiency. In this thesis, we develop new variational methods, studying their theoretical properties and applications.

In the first part of the thesis, we utilize dependence structures towards addressing fundamental problems in variational inference (VI): posterior uncertainty estimation, convergence properties, and discrete optimization. Though it is flexible, variational inference often underestimates the posterior uncertainty.

This is a consequence of the over-simplified variational family. Mean-field variational inference (MFVI), for example, uses a product of independent distributions as a coarse approximation to the posterior. As a remedy, we propose a hierarchical variational distribution with flexible parameterization that can model the dependence structure between latent variables. With a newly derived objective, we show that the proposed variational method can achieve accurate and efficient uncertainty estimation.

We further theoretically study the structured variational inference in the setting of the Stochastic Blockmodel (SBM). The variational distribution is constructed with a pairwise structure among the nodes of a graph. We prove that, in a broad density regime and for general random initializations, the estimated class labels by structured VI converge to the ground truth with high probability. Empirically, we demonstrate structured VI is more robust compared with MFVI when the graph is sparse and the signal to noise ratio is low.

When the latent variables are discrete, gradient descent based VI often suffers from bias and high variance in the gradient estimation. With correlated random samples, we propose a novel unbiased, low-variance gradient estimator. We demonstrate that under certain constraints, such correlated sampling gives an optimal control variates for the variance reduction. The efficient gradient estimation can be applied to solve a wide range of problems such as the variable selection, reinforcement learning, natural language processing, among others.

For the second part of the thesis, we apply variational methods to

the study of generalization problems in the meta-learning. When trained over multiple-tasks, we identify that a variety of the meta-learning algorithms implicitly require the tasks to have a mutually-exclusive dependence structure. This prevents the task-level overfitting problem and ensures the fast adaptation of the algorithm in the face of a new task. However, such dependence structure may not exist for general tasks. When the tasks are non-mutually exclusive, we develop new meta-learning algorithms with variational regularization to prevent the task-level overfitting. Consequently, we can expand the meta-learning to the domains which it cannot be effective on before.

Attribution This dissertation incorporates the outcomes from extensive collaborations. Chapter 2 for uncertainty estimation is the product of collaboration with Dr. Mingyuan Zhou and was published at International Conference on Machine Learning (ICML) 2018. Chapter 3 pertains to the theoretical analysis of structured VI which was completed with Doctors Y. X. Rachel Wang and Purnamrita Sarkar, and was published at International Conference on Artificial Intelligence and Statistics (AISTATS) 2020. Chapter 4 includes the discrete optimization work with Dr. Mingyuan Zhou and was published at International Conference on Learning Representations (ICLR) 2019. Chapter 5 for meta-learning is the result of collaboration with Doctors George Tucker, Mingyuan Zhou, Sergey Levine and Chelsea Finn, published at ICLR 2020.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Et toute science [...] déploie au cours des générations et des siècles, par le délicat contrepoint de tous les thèmes apparus tour à tour, comme appelés du néant, pour se joindre en elle et s'y entrelacer.

– Alexander Grothendieck, *Récoltes et Semailles, 1986*

The idea of using unobserved concepts to explain the observed phenomena can date back as far as to the ancient religions. In statistics, this approach is formalized scientifically as the latent variable models (LVM). Latent variables provide a flexible way to model the data generating mechanism and incorporate the unobserved quantities of interest. Extracted information about latent variables, such as the point and uncertainty estimations, can help describe observations and analyze the substream problems.

1.1 Variational Inference

In Bayesian statistics, the inference of latent variables is framed as a calculation of the posterior distribution, which meets both opportunities and challenges in this information age. We have witnessed a dramatic increase in the scale of data and dimension of variables, which make it possible to build

complex statistical models but often make it intractable to compute the exact posterior distribution. The motivation of this thesis is to find efficient and accurate methods for the approximate posterior inference and apply them to the new domains.

For decades, Markov chain Monte Carlo (MCMC) has achieved great success in approximate inference, which is based on random sampling. It approximates posterior with correlated samples, where a subsequent sample is drawn conditional on its previous one. When the ergodic condition is satisfied and the stationary distribution matches the posterior, these random samples can be considered as coming from the posterior distribution asymptotically. Though enjoy theoretical guarantees in an ideal setting, MCMC methods often meet challenges in practice such as slow mixing, sensitivity to initialization and requiring problem-specific design. These challenges are amplified when dealing with large scale data and hence call for alternative approaches.

Variational inference (VI) is an efficient and flexible paradigm for posterior inference, which is based on the optimization. It is widely believed to make up for the deficiencies in MCMC. To introduce the concepts of VI, we first define a general latent variable model as a joint density

$$p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}) \tag{1.1}$$

where $\boldsymbol{x} = \{\boldsymbol{x}_i\}_{i=1}^n$ are observed data and $\boldsymbol{z} = \{\boldsymbol{z}_j\}_{j=1}^m$ are latent variables. Variational inference specifies a family of distribution $\mathcal{Q}$ and finds the one within the family that is closest to the exact posterior, where the closeness is

measured by certain divergence $\mathcal{D}$

$$q^*(\boldsymbol{z}) = \arg\min_{q \in \mathcal{Q}} \ \mathcal{D}(q(\boldsymbol{z})||p(\boldsymbol{z}|\boldsymbol{x})) \tag{1.2}$$

Since exact posterior is assumed unknown, Eq. (1.2) cannot be optimized directly. When the divergence $\mathcal{D}$ is chosen as the Kullback-Leibler (KL) divergence, defined as $\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z})||p(\boldsymbol{z})) = \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z})} \log[q(\boldsymbol{z})/p(\boldsymbol{z})]$, the log-likelihoood of data (evidence) can be decomposed as

$$\log[p(\boldsymbol{x})] = \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z})}\log[p(\boldsymbol{x},\boldsymbol{z})/q(\boldsymbol{z})] + \mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z})||p(\boldsymbol{z}|\boldsymbol{x})) \tag{1.3}$$

With the identity above, instead of minimizing $\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z})||p(\boldsymbol{z}|\boldsymbol{x}))$, we can maximize the first term on the right hand side as an equivalent objective, known as the evidence lower bound (ELBO)

$$\mathcal{L}(q) = \mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z})}\log[p(\boldsymbol{x},\boldsymbol{z})/q(\boldsymbol{z})]. \tag{1.4}$$

The $q^*(\boldsymbol{z})$ that maximizes ELBO is a good approximation to the poterior within family $\mathcal{Q}$.

To maximize the ELBO, we need to specify the variational family $\mathcal{Q}$ and the optimization rule. MFVI, for example, chooses a factorized variational family as $q(\boldsymbol{z}) = \prod_{j=1}^{m} q(\boldsymbol{z}_j)$, which assumes independence between latent variables. Moreover, each factor $q(\boldsymbol{z}_j)$ is often set as an exponential family distribution. Given the local conjugacy, such independence and exponential family assumptions make closed-form coordinate ascent feasible. Though computationally efficient, MFVI is known for its inaccurate uncertainty estimation and

sensitivity to the random initializations. Modern variational methods expand the variational family by incorporating dependence structures between latent variables and change optimize rule to the gradient ascent. However, the answer remains elusive on how to accurately estimate the posterior uncertainty for VI, whether the dependence structure theoretically improves the convergence properties and how to efficiently estimate the gradient for the distribution parameters when latent variables are discrete. This thesis provides new insights towards answering these questions.

In Chaper 2, we introduce a flexible hierarchical distribution that can have millions of parameters constructed by deep neural networks. The universal approximation theory of neural network provides desired flexibility to approximate the posterior with high accuracy. Specifically, we introduce an auxiliary random variable $\boldsymbol{\psi}$ that is the output of a deterministic transformation with parameters $\boldsymbol{\phi}$. We use the marginal of the hierarchical model as the variational family

$$\mathcal{Q} = \{q_{\boldsymbol{\phi}}(\boldsymbol{z}) = \int q(\boldsymbol{z}|\boldsymbol{\psi}) q_{\boldsymbol{\phi}}(\boldsymbol{\psi})\ d\boldsymbol{\psi}\} \tag{1.5}$$

Here, the conditional distribution $q(\boldsymbol{z}|\boldsymbol{\psi})$ is required to be a simple explicit distribution such as Gaussian distribution, while $q(\boldsymbol{\psi})$ is allowed to be flexible without analytic density, as long as it can generate random samples. We call distribution in form of (1.5) as the semi-implicit distribution. The punchline is that the marginal distribution $q(\boldsymbol{z})$ can encode the dependency structure between the elements of $\boldsymbol{z}$ therefore can improve the uncertainty estimation.

However, $q(\boldsymbol{z})$ is not analytic in general so we cannot optimize the ELBO directly. To cope with the computational challenge, we derive a novel surrogate ELBO as the alternative objective function and theoretically prove that it convergences to the ELBO monotonically and asymptotically. We show that the semi-implicit distribution, which combines the explicit and implicit distributions, the deterministic and stochastic transformations, the conditional independence and marginal dependence, can achieve efficient and accurate posterior estimation.

To further understand the contribution of dependence structure in improving variational inference, in Chapter 3 we theoretically study the convergence properties of the structured VI and compare them with MFVI. We study the community detection problem with Stochastic Blockmodel and design a simplified pairwise dependence structure between the graph nodes. We prove that in a broad density regime and under a fairly general random initialization scheme, the pairwise structured VI can converge to the ground truth with probability tending to one when the parameters are known, estimated within a reasonable range, or updated appropriately. This is in contrast to MFVI, where convergence only happens for a narrower range of initializations. In addition, pairwise structured VI can escape from certain local optima that exist in the MFVI objective. These results highlight the theoretical advantage of the dependence structure.

When optimized with a stochastic gradient, the bias and variance of the gradient estimation influence the optima that variational methods can

converge to. It becomes salient when the latent variables are discrete, which in practice, are widely used in clustering, natural language models, variable selection, among others. This motivates our research in Chapter 4, so as to find unbiased and low variance gradient estimator for high dimensional discrete variables. We study an optimization objective in the form of $\mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z}|\boldsymbol{\phi})}[f(\boldsymbol{z})]$ where $\boldsymbol{z}$ is a vector of Bernoulli variables and $\boldsymbol{\phi}$ are the parameters. This objective has variational inference as a special case, as it becomes the ELBO when $f(\boldsymbol{z}) = \log[p(\boldsymbol{x}, \boldsymbol{z})/q(\boldsymbol{z})]$. To construct a gradient estimator, we first transform the expectation over Bernoulli variables to be over Uniform variables. Utilizing the symmetric property of the uniform distribution, we draw a pair of dependent samples to estimate the gradient at each iteration. The proposed estimation is unbiased, low-variance and has minimal computational cost. We prove that, under certain assumptions, such correlated sampling produces the optimal control variates.

1.2 Variational Methods for Statistical Learning

Variational methods are widely applied to the problems where the primary goals are to learn the model parameters. In these scenarios, the latent variables are introduced to construct the statistical models. We consider the data likelihood

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \log \int p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z} \tag{1.6}$$

with data $\boldsymbol{x} = \{\boldsymbol{x}_i\}_{i=1}^n$, latent variables $\boldsymbol{z} = \{\boldsymbol{z}_j\}_{j=1}^m$ and model parameters $\boldsymbol{\theta}$. The goal is to learn parameter $\boldsymbol{\theta}$ that maximizes the likelihood function,

known as the maximum likelihood estimation (MLE).

1.2.1 Expectation-Maximization Algorithm

The learning and inference problems are coupled. Pioneer variational methods can date back to the Expectation-Maximization (EM) algorithm [26]. The EM algorithm maximizes the likelihood by iteratively computing and maximizing a lower bound, which has the form as

$$\boldsymbol{\theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} \ \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{\theta}^{(t)}}(\boldsymbol{z}|\boldsymbol{x})} \log[p_{\boldsymbol{\theta}}(\boldsymbol{x},\boldsymbol{z})/p_{\boldsymbol{\theta}^{(t)}}(\boldsymbol{z}|\boldsymbol{x})]. \tag{1.7}$$

The expectation step of EM computes the ELBO based on the model parameters estimation at step t; then the maximization step sets the model parameters that maximize the ELBO as the updated estimation at step $t+1$. EM algorithm can be considered as a variational method which uses the exact posterior $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$ as the variational distribution. Such exact inference makes the ELBO tight, which equal to the evidence at the point where the posterior is computed. As a result, EM algorithm is monotonically nondecreasing in estimating the likelihood, i.e. $p_{\boldsymbol{\theta}_{t+1}}(\boldsymbol{x}) \geq p_{\boldsymbol{\theta}_t}(\boldsymbol{x})$. In the scenarios when the exact posterior is unknown, there is a gap between evidence and ELBO which can be quantified as $\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z})||p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x}))$. Accurate inference of posterior can reduce this gap and hence improve the learning of model parameters.

1.2.2 Deep Generative Model

Variational Autoencoder (VAE) [69] is a generative model that simulates the data generating process with deep neural networks. VAE is trained by

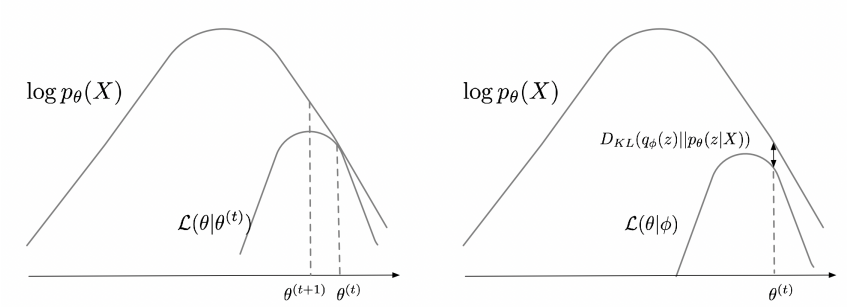


Figure 1.1: Demonstration of the statistical inference and learning based on the bound optimization. The left and right panel corresponds to the exact and approximate inference respectively.

maximizing the marginal likelihood of data. Based on the variational principles, the optimization objective is the ELBO

$$\max_{\boldsymbol{\phi},\boldsymbol{\theta}} \ \mathbb{E}_{\boldsymbol{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \log[p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})/q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})] \tag{1.8}$$

The $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ is called the encoder and $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ is called the decoder, which are both modeled by deep neural networks, with parameters $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ respectively. VAE iteratively optimizes the encoder and decoder parameters. The logic is that by optimizing the encoder, the variational distribution gets close to the postrior, which can reduce the gap between the lower bound and the data log-likelihood. The tightened bound can then improve the accuracy in learning decode parameters (see Figure 1.1). With high flexibility, VAEs are widely used in generating images, natural languages, molecular structures, and so on. In this thesis, we improve the quality of samples generated by VAE and learn VAE with discrete latent variables.

1.2.3 Multi-task Learning

In standard supervised learning, an algorithm is designed to solve a particular task. An intelligent system, however, is expected to have versatility. We expect a learning agent, having trained on multiple related tasks, can solve new tasks at the test-time efficiently by leveraging the past experience. To achieve such efficient generalization across tasks, meta-learning is a promising paradigm. In concordance with the meta-learning nomenclature, we assume correlated tasks $\mathcal{T}_i$ are sampled from a distribution $p(\mathcal{T})$. For each task, we observe a set of training data $\mathcal{D}_i = (\boldsymbol{x}_i, \boldsymbol{y}_i)$ and a set of test data $\mathcal{D}_i^* = (\boldsymbol{x}_i^*, \boldsymbol{y}_i^*)$ with $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{iK})$, $\boldsymbol{y}_i = (y_{i1}, \ldots, y_{iK})$ sampled from $p(x, y|\mathcal{T}_i)$, and similarly for $\mathcal{D}_i^*$. We denote $X^* = \{\boldsymbol{x}_i^*\}_{i=1}^N$, $Y^* = \{\boldsymbol{y}_i^*\}_{i=1}^N$. Our goal is to maximize the predictive likelihood for labels which is the marginal of a latent variable model

$$\log p(Y^*|X^*, \{\mathcal{D}_i\}_{i=1}^N) = \frac{1}{N}\sum_{i=1}^{N} \log\Big(\int p_{\boldsymbol{\theta}}(\boldsymbol{y}_i^*|\boldsymbol{x}_i^*, \boldsymbol{\phi}_i) p_{\boldsymbol{\theta}}(\boldsymbol{\phi}_i|\mathcal{D}_i) d\boldsymbol{\phi}_i\Big) \tag{1.9}$$

The variables ϕ_i are latent variables that summarize task-specific information and $\boldsymbol{\theta}$ are the model parameters to learn.

In Chaper 5, we identify two types of local optima in the landscape of meta-learning objective (1.9): one can be reached by adapting to the task training data $\mathcal{D}$ and the other can be reached by memorizing the task identities during meta training. The former solution is desired so that meta-learner can achieve the fast adaptation. The latter solution, however, is a task-level overfitting problem. We call it the memorization problem.

For example in personalized medicine, an ideal automated medical system can suggest medication prescriptions to doctors based on the symptoms, patients' identity information, and also adapt to the patients' individual medical history. In the meta-learning framework, each patient represents a separate task. A standard meta-learning system can memorize the patients' identity information, leading it to ignore the medical history and only utilize the symptoms combined with the memorized information. As a result, it can issue highly accurate prescriptions for the meta-training patients, but fail to use the personalized medical history to adapt to the new patients at the test-time.

Inspired by the variational information bottleneck, we propose a meta-regularization approach to address the memorization problem. Intuitively, with an objective encouraging low training error and low information stored in the meta-parameters, it forces the meta-learner to use the task training data to make predictions, therefore it favors the adaptation solution to the memorization solution.

In the remainder of this thesis, we will present new variational methods and theories for statistical inference and learning. Though scattered in different chapters, the analysis is nevertheless motivated by the dependence structures that inherently exist in random samples, random variables, and random functions. More generally, it echos the philosophy of naturalist John Muir, "when we try to pick out anything by itself, we find it hitched to everything else in the universe."

Chapter 2

Uncertainty Estimation in Variational Inference

This chapter is devoted to uncertainty estimation in variational inference, based on publication [161]. To achieve accurate posterior approximation, we introduce semi-implicit variational inference (SIVI) to expand the commonly used analytic variational family, by mixing the variational parameter with a flexible distribution. This mixing distribution can assume any density function, explicit or not, as long as independent random samples can be generated via reparameterization. We derive a new optimization objective as a surrogate evidence lower bound (ELBO). The tightness of the bound is demonstrated by the asymtotic and monotonic properties. With a substantially expanded variational family and a novel optimization algorithm, SIVI closely matches the accuracy of MCMC in inferring the posterior in a variety of Bayesian inference tasks.

The content in this chapter was published in [161], Yin, Mingzhang and Mingyuan Zhou. "Semi-Implicit Variational Inference". In International Conference on Machine Learning, pp. 5646-5655. 2018. I designed the algorithm with Prof. Zhou, proved the theoretical properties, implemented the simulations and wrote the draft. Prof. Zhou proposed the problem, proposed the initial methodology, brainstormed about the experimental setting, helped with the draft rewriting and revising.

2.1 Variational Inference with Dependence Structures

Variational inference (VI) is an optimization based method that is widely used for approximate Bayesian inference. Despite its popularity, VI has a well-known issue in underestimating the variance of the posterior, which is often attributed to the mismatch between the representation power of the variational family that Q is restricted to and the complexity of the posterior. This issue is often further amplified in mean-field VI (MFVI), due to the factorized assumption on Q that ignores the dependencies between different factorization components [151, 16].

There exists a wide variety of VI methods that improve on MFVI by modeling dependence structures in latent variables. A simple but powerful approach is to construct joint ditribution of latent variables by complex deterministic and/or stochastic transformations. One successful application of this idea in VI is constructing the variational distribution with a normalizing flow, which transforms a simple random variable through a sequence of invertible differentiable functions with tractable Jacobians, to deterministically map a simple PDF to a complex one [116, 70, 103].

Normalizing flows help increase the flexibility of VI, but still require the mapping to be deterministic and invertible. Removing both restrictions, there have been several recent attempts to define highly flexible variational distributions with implicit model [59, 92, 146, 79, 88, 130]. A typical example is transforming random noise via a deep neural network, leading to a non-invertible highly nonlinear mapping and hence an implicit distribution. While an implicit

variational distribution can be made highly flexible, it becomes necessary in each iteration to address the problem of density ratio estimation, which is often transformed into a problem related to learning generative adversarial networks [41]. In particular, a binary classifier, whose class probability is used for density ratio estimation, is trained in each iteration to discriminate the samples generated by the model from those by the variational distribution [92, 148, 88]. Controlling the bias and variance in density ratio estimation, however, is in general a very difficult problem, especially in high-dimensional settings [134].

Besides deteministic transformation, there are a variety of algorithms for structured VI. Examples include modeling dependence between local and global parameters [127, 55], using a mixture of variational distributions [14, 37, 122, 89], introducing a copula to capture the dependencies between univariate marginals [145, 52], handling non-conjugacy [102, 142], and constructing a hierarchical variational distribution [113, 146, 82, 2].

To well characterize the posterior while maintaining simple optimization, we introduce semi-implicit VI (SIVI) that imposes a mixing distribution on the parameters of Q to expand the variational family with a semi-implicit hierarchical construction. The meaning of "semi-implicit" is twofold: 1) the original Q distribution is required to have an analytic PDF, but its mixing distribution is not subject to such a constraint; and 2) even if both the original Q and its mixing distribution have analytic PDFs, it is common that the marginal is implicit, that is, having a non-analytic PDF. Our intuition behind

SIVI is that even if the marginal variational distribution is not tractable, its density can be evaluated with Monte Carlo estimation under this semi-implicit hierarchical construction, an expansion that helps model skewness, kurtosis, multimodality, and other characteristics that are exhibited by the posterior but failed to be captured by the original variational family. For MFVI, an evident benefit of this expansion is restoring the dependencies between its factorization components, as the resulted Q distribution becomes conditionally independent but marginally dependent.

SIVI makes three major contributions: 1) a reparameterizable implicit distribution can be used as a mixing distribution to effectively expand the richness of the variational family; 2) an analytic conditional Q distribution is used to sidestep the hard problem of density ratio estimation, and is not required to be reparameterizable in conditionally conjugate models; and 3) SIVI sandwiches the ELBO between a lower bound and an upper bound, and derives an asymptotically exact surrogate ELBO that is amenable to direct optimization via stochastic gradient ascent. With a flexible variational family and novel optimization, SIVI bridges the accuracy gap of posterior estimation between VI and Markov chain Monte Carlo (MCMC), which can accurately characterize the posterior using MCMC samples, as will be demonstrated in a variety of Bayesian inference tasks.

2.2 Semi-Implicit Variational Inference

In VI, given observations $\boldsymbol{x}$, latent variables $\boldsymbol{z}$, model likelihood $p(\boldsymbol{x} \mid \boldsymbol{z})$, and prior $p(\boldsymbol{z})$, we approximate the posterior $p(\boldsymbol{z} \mid \boldsymbol{x})$ with variational distribution $q(\boldsymbol{z} \mid \boldsymbol{\psi})$ that is often required to be explicit. We optimize the variational parameter $\boldsymbol{\psi}$ to mximize the evidence lower bound (ELBO) as Eq. (1.4)

$$\text{ELBO} = \mathbb{E}_{z \sim q(\boldsymbol{z}|\boldsymbol{\psi})} \log[p(\boldsymbol{x}, \boldsymbol{z})/q(\boldsymbol{z}|\boldsymbol{\psi})] \tag{2.1}$$

Rather than treating $\boldsymbol{\psi}$ as the variational parameter to be inferred, SIVI regards $\boldsymbol{\psi} \sim q(\boldsymbol{\psi})$ as a random variable. Assuming $\boldsymbol{\psi} \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$, where $\boldsymbol{\phi}$ denotes the distribution parameter to be inferred, the semi-implicit variational distribution for $\boldsymbol{z}$ can be defined in a hierarchical manner as $\boldsymbol{z} \sim q(\boldsymbol{z} \mid \boldsymbol{\psi}),\ \boldsymbol{\psi} \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$. Marginalizing the intermediate variable $\boldsymbol{\psi}$ out, we can view $\boldsymbol{z}$ as a random variable drawn from distribution family $\mathcal{H}$ indexed by variational parameter $\boldsymbol{\phi}$, expressed as

$$\mathcal{H} = \left\{ h_{\boldsymbol{\phi}}(\boldsymbol{z}) :\ h_{\boldsymbol{\phi}}(\boldsymbol{z}) = \textstyle\int_{\boldsymbol{\psi}} q(\boldsymbol{z} \mid \boldsymbol{\psi}) q_{\boldsymbol{\phi}}(\boldsymbol{\psi}) d\boldsymbol{\psi} \right\}.$$

Note $q(\boldsymbol{z} \mid \boldsymbol{\psi})$ is required to be explicit, but the mixing distribution $q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$ is allowed to be implicit. Moreover, unless $q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$ is conjugate to $q(\boldsymbol{z} \mid \boldsymbol{\psi})$, the marginal Q distribution $h_{\boldsymbol{\phi}}(\boldsymbol{z}) \in \mathcal{H}$ is often implicit. These are the two reasons for referring to the proposed VI as semi-implicit VI (SIVI).

SIVI requires $q(\boldsymbol{z} \mid \boldsymbol{\psi})$ to be explicit, and also requires it to either be reparameterizable, which means $\boldsymbol{z} \sim q(\boldsymbol{z} \mid \boldsymbol{\psi})$ can be generated by transforming random noise $\boldsymbol{\varepsilon}$ via function $f(\boldsymbol{\varepsilon}, \boldsymbol{\psi})$, or allow ELBO to be analytic. Whereas the

mixing distribution $q(\boldsymbol{\psi})$ is required to be reparameterizable but not necessarily explicit. In particular, SIVI draws from $q(\boldsymbol{\psi})$ by transforming random noise $\boldsymbol{\epsilon}$ via a deep neural network, which generally leads to an implicit distribution for $q(\boldsymbol{\psi})$ due to a non-invertible transform.

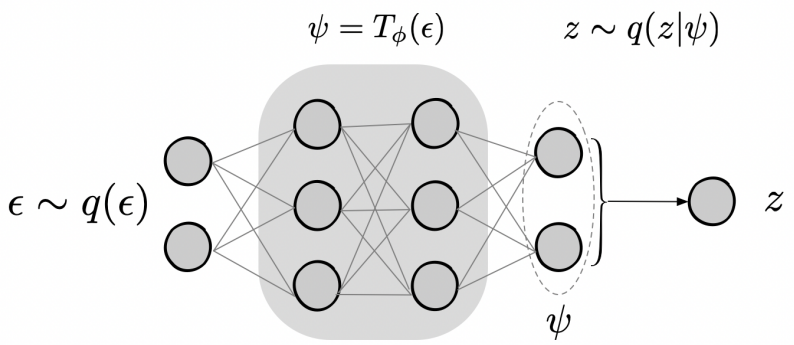


Figure 2.1: Demonstration of sampling from semi-implicit distribution.

While restricting $q(\boldsymbol{z} \mid \boldsymbol{\psi})$ to be explicit, SIVI introduces a mixing distribution $q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$ to enhance its representation power. In this paper, we construct $q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$ with an implicit distribution that generates its random samples via a stochastic procedure but may not allow a pointwise evaluable PDF. More specifically, an implicit distribution [92, 146], consisting of a source of randomness $q(\boldsymbol{\epsilon})$ for $\boldsymbol{\epsilon} \in \mathbb{R}^g$ and a deterministic transform $T_{\boldsymbol{\phi}} : \mathbb{R}^g \to \mathbb{R}^d$, can be constructed as $\boldsymbol{\psi} = T_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}),\ \boldsymbol{\epsilon} \sim q(\boldsymbol{\epsilon})$, with PDF

$$q_{\boldsymbol{\phi}}(\boldsymbol{\psi}) = \frac{\partial}{\partial \psi_1} \cdots \frac{\partial}{\partial \psi_d} \int_{T_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}) \le \boldsymbol{\psi}} q(\boldsymbol{\epsilon}) d\boldsymbol{\epsilon}. \tag{2.2}$$

When $T_{\boldsymbol{\phi}}$ is invertible and the integration is tractable, the PDF of $\boldsymbol{\psi}$ can be calculated with (2.2), but this is not the case in general and hence $q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$ is

often implicit. When $T_{\boldsymbol{\phi}}(\cdot)$ is chosen as a deep neural network, thanks to its high modeling capacity, $q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$ can be highly flexible and the dependencies between the elements of $\boldsymbol{\psi}$ can be well captured.

Prevalently used in the study of thermodynamics, ecology, epidemiology, and differential equation systems, implicit distributions have only been recently introduced in VI to parameterize $q(\boldsymbol{z} \mid \boldsymbol{\psi})$ [79, 88, 59, 146]. Using implicit distributions with intractable PDF increases flexibility but substantially complicates the optimization problem for VI, due to the need to estimate log density ratios involving intractable PDFs, which is particularly challenging in high dimensions [134]. By contrast, taking a semi-implicit construction, SIVI offers the best of both worlds: constructing a highly flexible variational distribution, without sacrificing the key benefit of VI in converting posterior inference into an optimization problem that is simple to solve. Below we develop a novel optimization algorithm that exploits SIVI's semi-implicit construction.

2.3 Optimization for SIVI

To optimize the variational parameters of SIVI, below we first derive for the ELBO a lower bound, climbing which, however, could drive the mixing distribution $q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$ towards a point mass density. To prevent degeneracy, we add a nonnegative regularization term, leading to a surrogate ELBO that is monotonic and asymptotically exact, as can be further tightened by importance reweighting. To derive a tractable optimization objective, we first show the convexity of KL divergence

Theorem 1 ([23]). *The KL divergence from distribution $q(\boldsymbol{z})$ to $p(\boldsymbol{z})$, expressed as $\mathcal{D}_{KL}(q(\boldsymbol{z})||p(\boldsymbol{z}))$, is convex in the pair $(q(\boldsymbol{z}), p(\boldsymbol{z}))$.*

Fixing the distribution $p(\boldsymbol{z})$ in Theorem 1, the KL divergence can be viewed as a convex functional in $q(\boldsymbol{z})$. As in Appendix A, extending Jensen's inequality leads to

$$\mathcal{D}_{\mathrm{KL}}(\mathbb{E}_{\boldsymbol{\psi}} q(\boldsymbol{z} \mid \boldsymbol{\psi})||p(\boldsymbol{z})) \leq \mathbb{E}_{\boldsymbol{\psi}} \mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z} \mid \boldsymbol{\psi})||p(\boldsymbol{z})). \tag{2.3}$$

Substituting $h_{\boldsymbol{\phi}}(\boldsymbol{z}) = \mathbb{E}_{\boldsymbol{\psi} \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})} q(\boldsymbol{z} \mid \boldsymbol{\psi})$ and $p(\boldsymbol{x}, \boldsymbol{z})$ into (2.3) leads to a lower bound of SIVI's ELBO $\mathcal{L}$ as

$$\underline{\mathcal{L}}(q(\boldsymbol{z} \mid \boldsymbol{\psi}), q_{\boldsymbol{\phi}}(\boldsymbol{\psi})) = \mathbb{E}_{\boldsymbol{\psi} \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})} \mathbb{E}_{\boldsymbol{z} \sim q(\boldsymbol{z} \mid \boldsymbol{\psi})} \log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{q(\boldsymbol{z} \mid \boldsymbol{\psi})} \leq \mathcal{L} \tag{2.4}$$

A Monte Carlo estimation of $\underline{\mathcal{L}}$ only requires $q(\boldsymbol{z} \mid \boldsymbol{\psi})$ to have an analytic PDF and $q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$ to be convenient to sample from. It is this nice separation of evaluation and sampling that allows SIVI to combine an explicit $q(\boldsymbol{z} \mid \boldsymbol{\psi})$ with an implicit $q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$ that is as powerful as needed, while maintaining tractable computation.

2.3.1 Degeneracy Problem

A direct optimization of the lower bound $\underline{\mathcal{L}}$ in (2.4), however, can suffer from degeneracy, as shown in the proposition blow. All proofs are deferred to Appendix A.

Proposition 1. *Let us denote $\boldsymbol{\psi}^* = \arg\max_{\boldsymbol{\psi}} -\mathcal{D}_{KL}(q(\boldsymbol{z} \mid \boldsymbol{\psi})||p(x, \boldsymbol{z}))$, then*

$$\underline{\mathcal{L}}(q(\boldsymbol{z} \mid \boldsymbol{\psi}), q_{\boldsymbol{\phi}}(\boldsymbol{\psi})) \leq -\mathcal{D}_{KL}(q(\boldsymbol{z} \mid \boldsymbol{\psi}^*)||p(x, \boldsymbol{z})),$$

where the equality is true if and only if $q_{\boldsymbol{\phi}}(\boldsymbol{\psi}) = \delta_{\boldsymbol{\psi}^*}(\boldsymbol{\psi})$.

Therefore, if optimizing the variational parameter on $\underline{\mathcal{L}}(q(\boldsymbol{z} \mid \boldsymbol{\psi}), q_{\boldsymbol{\phi}}(\boldsymbol{\psi}))$, without stopping the optimization algorithm early, $q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$ could converge to a point mass density, making SIVI degenerate to vanilla VI.

2.3.2 Surrogate Lower Bound

To prevent degeneracy, we regularize $\underline{\mathcal{L}}$ by adding

$$I_K(\boldsymbol{\phi}) = \mathbb{E}_{\boldsymbol{\psi}, \boldsymbol{\psi}^{(1)}, \ldots, \boldsymbol{\psi}^{(K)} \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})} \mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z} \mid \boldsymbol{\psi}) || \tfrac{1}{K+1}[q(\boldsymbol{z} \mid \boldsymbol{\psi}) + \textstyle\sum_{k=1}^{K} q(\boldsymbol{z} \mid \boldsymbol{\psi}^{(k)})].). \quad (2.5)$$

Clearly, $I_K \geq 0$, with $I_K = 0$ if and only if $K = 0$ or $q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$ degenerates to a point mass density. Therefore, $\underline{\mathcal{L}}_0 = \underline{\mathcal{L}}$ and maximizing $\underline{\mathcal{L}}_K = \underline{\mathcal{L}} + I_K$ with $K \geq 1$ would encourage positive I_K and drive $q(\boldsymbol{\psi})$ away from degeneracy. Combining (2.4) and (2.5), we have the final objective as SIVI-ELBO

$$\underline{\mathcal{L}}_K = \mathbb{E}_{\boldsymbol{\psi}, \boldsymbol{\psi}^{(1)}, \ldots, \boldsymbol{\psi}^{(K)} \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})} \mathbb{E}_{\boldsymbol{z} \sim q(\boldsymbol{z} \mid \boldsymbol{\psi})} \log \frac{p(\boldsymbol{x}, \boldsymbol{z})}{\frac{1}{K+1}\left[q(\boldsymbol{z} \mid \boldsymbol{\psi}) + \sum_{k=1}^{K} q(\boldsymbol{z} \mid \boldsymbol{\psi}^{(k)})\right]}. \quad (2.6)$$

In the following proposition, we show $\underline{\mathcal{L}}_K$ is indeed a lower bound of ELBO and thus a lower bound of evidence. We further show the bound is tightened monotonically as K increases and asymptotically converges to the ELBO. The proof of monotone property can be found in Molchanov et al. [93].

Proposition 2 (Lower Bound and monotonicity)**.** *Suppose* $\mathcal{L}$ *are defined as in* (2.1) *and* I_K *as in* (2.5)*, then* $\underline{\mathcal{L}}_K = \underline{\mathcal{L}} + I_K$ *monotonically converges from below towards the ELBO, satisfying* $\forall K$, $\underline{\mathcal{L}}_0 = \underline{\mathcal{L}}$, $\underline{\mathcal{L}}_K \leq \underline{\mathcal{L}}_{K+1} \leq \mathcal{L}$.

Moreover, as $\lim_{K\to\infty}\tilde{h}_K(\boldsymbol{z}) = \mathbb{E}_{\boldsymbol{\psi}\sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})} q(\boldsymbol{z}\,|\,\boldsymbol{\psi}) = h_{\boldsymbol{\phi}}(\boldsymbol{z})$ by the strong law of large numbers and

$$\lim_{K\to\infty} B_K = \mathbb{E}_{\boldsymbol{\psi}\sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})} \mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z}\,|\,\boldsymbol{\psi})||h_{\boldsymbol{\phi}}(\boldsymbol{z})) = I(\boldsymbol{z};\boldsymbol{\psi}) \tag{2.7}$$

by interchanging two limiting operations, as discussed in detail in Appendix A, we have the following proposition.

Proposition 3 (Asymptoticity)**.** $\lim_{K\to\infty}\underline{\mathcal{L}}_K = \mathcal{L}$

It is worth noting that the estimator I_K is a lower bound of mutual information $I(\boldsymbol{z};\boldsymbol{\psi})$ which measures the mutual dependence between latent variable $\boldsymbol{z}$ and auxilliary variable $\boldsymbol{\psi}$. This estimator was independently discovered by [100] as Noise-Contrastive Estimation (NCE).

The bound in (2.6) provides an information-theoretic decomposition of uncertainty estimation in variational inference

$$\underline{\mathcal{L}}_K(\boldsymbol{\phi}) = \underbrace{\underbrace{\underbrace{\mathbb{E}_{\boldsymbol{z}\sim h_{\boldsymbol{\phi}}(\boldsymbol{z})} \log p(\boldsymbol{x}|\boldsymbol{z})}_{\text{Point estimate}} - \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{\psi})} D_{KL}(q(\boldsymbol{z}|\boldsymbol{\psi})||p(\boldsymbol{z}))}_{\text{MFVI estimation}} + I_K(\boldsymbol{\phi})}_{\text{SIVI estimation}}.$$

It describes the tradeoff between the computational cost and posterior inference accuracy. Using the importance reweighting idea, Burda et al. [19] provides a lower bound $\mathcal{L}^{\tilde{K}} \geq \text{ELBO}$ that monotonically converges from below to the evidence $\log\ p(\boldsymbol{x})$ as $\tilde{K}$ increases. Using the same idea, we may also tighten the SIVI-ELBO in (2.6) using

$$\underline{\mathcal{L}}_K^{\tilde{K}} = \mathbb{E}_{(\boldsymbol{z}_i,\boldsymbol{\psi}_i)_{1:\tilde{K}}\sim q(\boldsymbol{z},\boldsymbol{\psi})} \mathbb{E}_{\boldsymbol{\psi}^{(1:K)}\sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})} \log \frac{1}{\tilde{K}} \sum_{i=1}^{\tilde{K}} \frac{p(\boldsymbol{x},\boldsymbol{z}_i)}{\frac{1}{K+1}\left[q(\boldsymbol{z}_i\,|\,\boldsymbol{\psi}_i) + \sum_{k=1}^{K} q(\boldsymbol{z}_i\,|\,\boldsymbol{\psi}^{(k)})\right]},$$

for which $\lim_{K\to\infty} \underline{\mathcal{L}}_K^{\tilde{K}} = \mathcal{L}^{\tilde{K}} \geq$ ELBO and $\lim_{K,\tilde{K}\to\infty} \underline{\mathcal{L}}_K^{\tilde{K}} = \lim_{\tilde{K}\to\infty} \mathcal{L}^{\tilde{K}} =$ log $p(\boldsymbol{x})$. Using $\underline{\mathcal{L}}_{K_t}$ as the surrogate ELBO, where t indexes the number of iterations, $K_t \in \{0, 1, \ldots\}$, and $K_{t+1} \geq K_t$, we describe the stochastic gradient ascend algorithm to optimize the variational parameter in Algorithm 1, in which we further introduce $\boldsymbol{\xi}$ as the variational parameter of the conditional distribution $q_{\boldsymbol{\xi}}(\boldsymbol{z} \mid \boldsymbol{\psi})$ that is not mixed with another distribution. For Monte Carlo estimation in Algorithm 1, we use a single random sample for each $\boldsymbol{\psi}^{(k)}$, J random samples for $\boldsymbol{\psi}$, and a single sample of $\boldsymbol{z}$ for each sample of $\boldsymbol{\psi}$. We denote $\boldsymbol{z} = f(\boldsymbol{\varepsilon}, \boldsymbol{\xi}, \boldsymbol{\psi}),\ \boldsymbol{\varepsilon} \sim p(\boldsymbol{\varepsilon})$ as the reparameterization for $\boldsymbol{z} \sim q_{\boldsymbol{\xi}}(\boldsymbol{z} \mid \boldsymbol{\psi})$. As for $\boldsymbol{\xi}$, if $\boldsymbol{\xi} \neq \emptyset$, one may learn it as in Algorithm 1, set it empirically, or fix it at the value learned by another algorithm such as MFVI. In summary, SIVI constructs a flexible variational distribution by mixing a (potentially) implicit distribution with an explicit one, while maintaining tractable optimization via the use of an asymptotically exact surrogate ELBO.

2.4 Experimental Results

We implement SIVI for a range of inference tasks. The toy examples show SIVI captures skewness, kurtosis, and multimodality. A negative binomial model shows SIVI can accurately capture the dependencies between latent variables. A bivariate count distribution example shows for a conditionally conjugate model, SIVI can utilize a non-reparameterizable variational distribution, without being plagued by the high variance of score function gradient estimation. With Bayesian logistic regression, we demonstrate that SIVI can either

Algorithm 1 Semi-Implicit Variational Inference (SIVI)

input : Data $\{x_i\}_{1:N}$, joint likelihood $p(\boldsymbol{x}, \boldsymbol{z})$, explicit variational distribution $q_{\boldsymbol{\xi}}(\boldsymbol{z} \mid \boldsymbol{\psi})$, implicit layer neural network $T_{\boldsymbol{\phi}}(\epsilon)$ and source of randomness $q(\boldsymbol{\epsilon})$

output : Variational parameter $\boldsymbol{\xi}$ for the conditional distribution $q_{\boldsymbol{\xi}}(\boldsymbol{z} \mid \boldsymbol{\psi})$, variational parameter $\boldsymbol{\phi}$ for the mixing distribution $q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$

Initialize $\boldsymbol{\xi}$ and $\boldsymbol{\phi}$ randomly

while *not converged* **do**

  Set $\underline{L}_{K_t} = 0$, ρ_t and η_t as step sizes, and $K_t \geq 0$ as a non-decreasing integer; Sample $\boldsymbol{\psi}^{(k)} = T_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}^{(k)})$, $\boldsymbol{\epsilon}^{(k)} \sim q(\boldsymbol{\epsilon})$ for $k = 1, \ldots, K_t$; take subsample $\mathbf{x} = \{x_i\}_{i_1:i_M}$

  for $j = 1$ **to** J **do**

    Sample $\boldsymbol{\psi}_j = T_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}_j)$, $\boldsymbol{\epsilon}_j \sim q(\boldsymbol{\epsilon})$

    Sample $\boldsymbol{z}_j = f(\tilde{\boldsymbol{\epsilon}}_j, \boldsymbol{\xi}, \boldsymbol{\psi}_j)$, $\tilde{\boldsymbol{\epsilon}}_j \sim p(\boldsymbol{\varepsilon})$

    $\underline{L}_{K_t} = \underline{L}_{K_t} + \frac{1}{J}\big\{ \log \frac{1}{K_t+1}\big[\sum_{k=1}^{K_t} q_{\boldsymbol{\xi}}(\boldsymbol{z}_j \mid \boldsymbol{\psi}^{(k)}) + q_{\boldsymbol{\xi}}(\boldsymbol{z}_j \mid \boldsymbol{\psi}_j)\big] - \frac{N}{M} \log p(\mathbf{x} \mid \boldsymbol{z}_j) - \log p(\boldsymbol{z}_j)\big\}$

  end

  $\boldsymbol{\xi} = \boldsymbol{\xi} + \rho_t \nabla_{\boldsymbol{\xi}} \underline{L}_{K_t}\big(\{\boldsymbol{\psi}^{(k)}\}_{1,K}, \{\boldsymbol{\psi}_j\}_{1,J}, \{\boldsymbol{z}_j\}_{1,J}\big)$

  $\boldsymbol{\phi} = \boldsymbol{\phi} + \eta_t \nabla_{\boldsymbol{\phi}} \underline{L}_{K_t}\big(\{\boldsymbol{\psi}^{(k)}\}_{1,K}, \{\boldsymbol{\psi}_j\}_{1,J}, \{\boldsymbol{z}_j\}_{1,J}\big)$

end

work alone as a black-box inference procedure for correlated latent variables, or directly expand MFVI by adding a mixing distribution, leading to accurate uncertainty estimation on par with that of MCMC. Last but not least, moving beyond the canonical Gaussian based variational autoencoder (VAE), SIVI helps construct semi-implicit VAE to improve unsupervised feature learning and amortized inference.

2.4.1 Expressiveness of SIVI

We first show the expressiveness of SIVI by approximating various target distributions. As listed in Table 2.1, the conditional layer of SIVI is chosen to be as simple as an isotropic Gaussian (or log-normal) distribution $\mathcal{N}(\mathbf{0}, \sigma_0^2 \mathbf{I})$.

Table 2.1: Inference and target distributions for SIVI in synthetic example.

$h(\boldsymbol{z}) = \mathbb{E}_{\boldsymbol{\psi} \sim q(\boldsymbol{\psi})} q(\boldsymbol{z} \mid \boldsymbol{\psi})$	$p(\boldsymbol{z})$
$z \sim \mathcal{N}(\psi, 0.1)$, $\psi \sim q(\psi)$	$\text{Laplace}(z; \mu = 0, b = 2)$
	$0.3\mathcal{N}(z; -2, 1) + 0.7\mathcal{N}(z; 2, 1)$
$z \sim \text{Log-Normal}(\psi, 0.1)$, $\psi \sim q(\psi)$	$\text{Gamma}(z; 2, 1)$
$\boldsymbol{z} \sim \mathcal{N}\left(\boldsymbol{\psi}, \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}\right)$, $\boldsymbol{\psi} \sim q(\boldsymbol{\psi})$	$0.5\mathcal{N}(\boldsymbol{z}; -2, I) + 0.5\mathcal{N}(\boldsymbol{z}; 2, I)$
	$\mathcal{N}(z_1; z_2^2/4, 1)\mathcal{N}(z_2; 0, 4)$
	$0.5\mathcal{N}\left(\boldsymbol{z}; 0, \begin{bmatrix} 2 & 1.8 \\ 1.8 & 2 \end{bmatrix}\right) + 0.5\mathcal{N}\left(\boldsymbol{z}; 0, \begin{bmatrix} 2 & -1.8 \\ -1.8 & 2 \end{bmatrix}\right)$

The implicit mixing layer is a multilayer perceptron (MLP), with layer widths $[30, 60, 30]$ and a ten dimensional isotropic Gaussian noise as its input. We fix $\sigma_0^2 = 0.1$ and optimize the implicit layer to minimize $\mathcal{D}_{\text{KL}}(\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{\psi})} q(\boldsymbol{z} \mid \boldsymbol{\psi}) || p(\boldsymbol{z}))$. As shown in Figure 2.2, despite having a fixed purposely misspecified explicit layer, by training a flexible implicit layer, the random samples from which are illustrated in Figure 2.3, SIVI infers a sophisticated marginal variational distribution that accurately captures the skewness, kurtosis, and/or multimodality exhibited by the target one.

2.4.2 Negative Binomial Model

We consider a negative binomial (NB) distribution with the gamma and beta priors ($a = b = \alpha = \beta = 0.01$) as

$$x_i \overset{i.i.d.}{\sim} \text{NB}(r, p),\ r \sim \text{Gamma}(a, 1/b),\ p \sim \text{Beta}(\alpha, \beta),$$

for which the posterior $p(r, p \mid \{x_i\}_{1,N})$ is not tractable. MFVI, which uses $q(r, p) = \text{Gamma}(r; \tilde{a}, 1/\tilde{b})\text{Beta}(p; \tilde{\alpha}, \tilde{\beta})$ to approximate the posterior, notably underestimates the variance [171] . This caveat of MFVI motivates a semi-

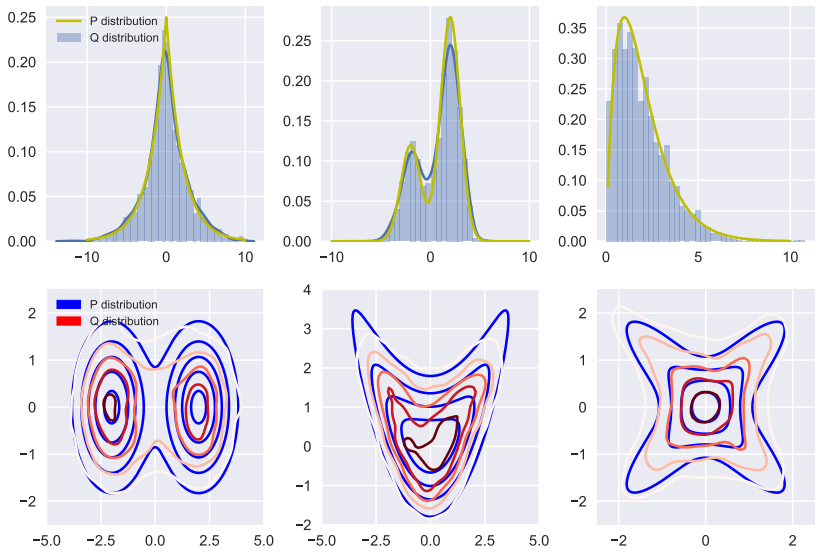


Figure 2.2: Approximating synthetic target distributions with SIVI

implicit variational distribution as

$$q(r, p \mid \boldsymbol{\psi}) = \text{Log-Normal}(r; \mu_r, \sigma_0^2) \cdot \text{Logit-Normal}(p; \mu_p, \sigma_0^2),$$
$$\boldsymbol{\psi} = (\mu_r, \mu_p) \sim q(\boldsymbol{\psi})$$

where an MLP based implicit $q(\boldsymbol{\psi})$, as in Section 2.4.1, is used by SIVI to capture the dependency between r and p.

We apply Gibbs sampling, MFVI, and SIVI to a real overdispersed count dataset of Bliss and Fisher [17] that records the number of adult red mites on each of the 150 randomly selected apple leaves. With $K = 1000$, as shown in Figure 2.4, SIVI clearly improves MFVI in closely matching the posterior inferred by Gibbs sampling. Moreover, the mixing distribution $q(\boldsymbol{\psi})$ clearly helps capture the negative correlation between r and p, as totally ignored by

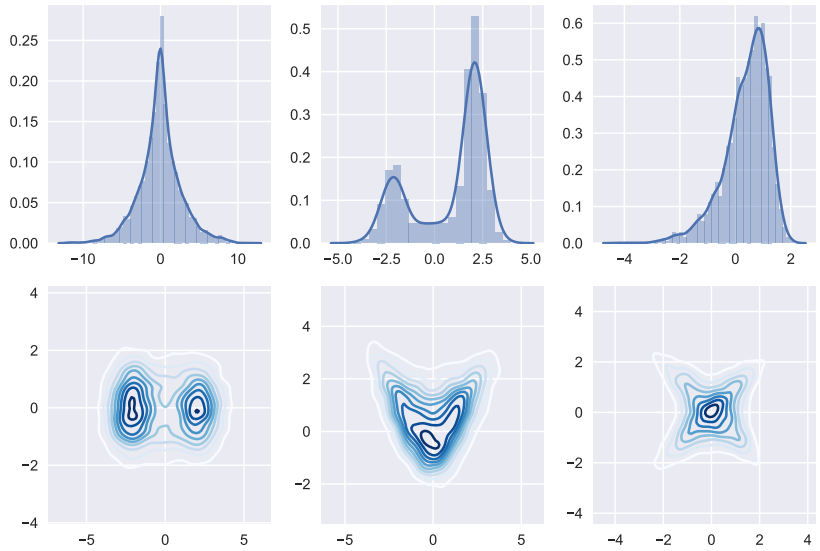


Figure 2.3: Visualization of the MLP based implicit distributions $\boldsymbol{\psi} \sim q(\boldsymbol{\psi})$.

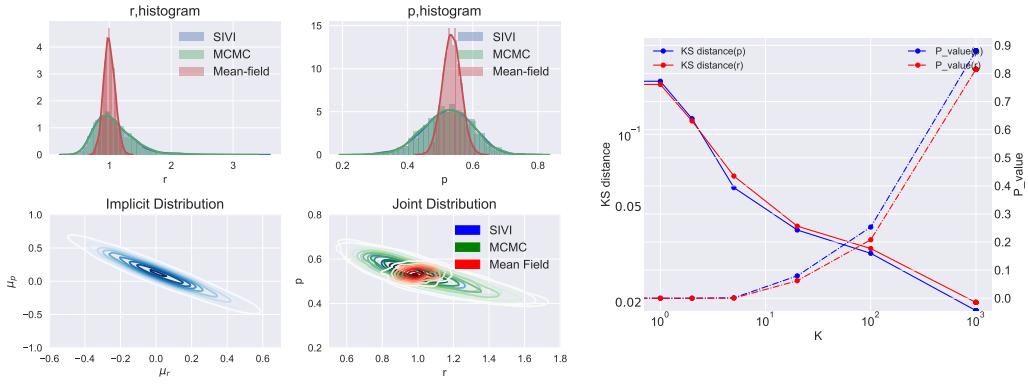


Figure 2.4: Top left row: the marginal posteriors of r and p inferred by MFVI, SIVI, and MCMC. Bottom left row: the inferred implicit mixing distribution $q(\boldsymbol{\psi})$ and joint posterior of r and p. Right: Kolmogorov-Smirnov (KS) distance and p-value between the marginal posteriors of r and p inferred by SIVI and MCMC.

MFVI. The two-sample Kolmogorov-Smirnov (KS) distances, between 2000 posterior samples generated by SIVI and 2000 MCMC ones, are 0.0185 (p-value = 0.88) and 0.0200 (p-value = 0.81) for r and p, respectively. By contrast, for MFVI and MCMC, they are 0.2695 (p-value = 5.26×10^{-64}) and 0.2965 (p-value = 2.21×10^{-77}), which significantly reject the null hypothesis that the posterior inferred by MFVI matches that by MCMC. As further suggested by Figure 2.4 and Figures 2.5, as K increases, the posterior inferred by SIVI quickly approaches that inferred by MCMC.

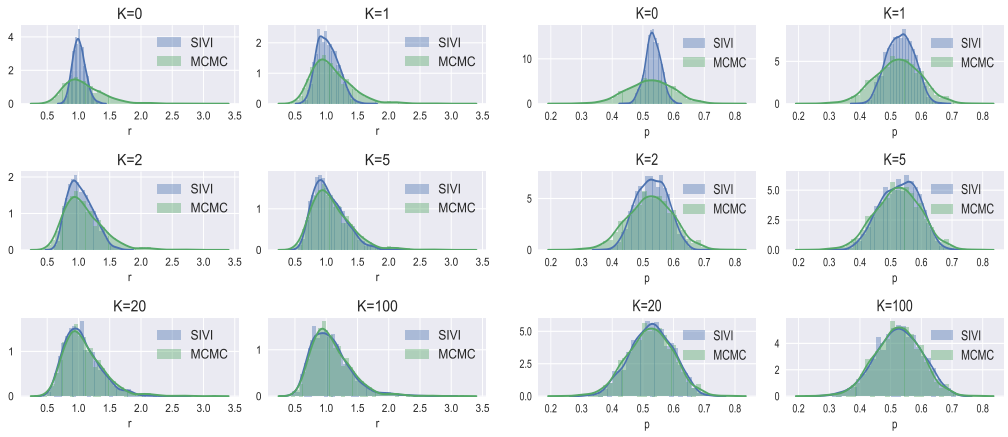


Figure 2.5: The marginal posterior distribution of the negative binomial probability parameter r and p inferred by SIVI.

2.4.3 Bayesian Logistic Regression

We compare SIVI with MFVI, Stein variational gradient descent (SVGD) of Liu and Wang [80], and MCMC on Bayesian logistic regression, expressed as

$$y_i \sim \text{Bernoulli}[(1 + e^{-\boldsymbol{x}_i'\boldsymbol{\beta}})^{-1}], \quad \boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \alpha^{-1}\mathbf{I}_{V+1}),$$

where $\boldsymbol{x}_i = (1, x_{i1}, \ldots, x_{iV})^T$ are covariate vectors, $y_i \in \{0, 1\}$ are binary response variables, and α is set as 0.01. With the Polya-Gamma data augmentation of Polson et al. [109], we collect posterior MCMC samples of $\boldsymbol{\beta}$ using a Gibbs sampling algorithm. For MFVI, the variational distribution is chosen as a multivariate normal (MVN) $\mathcal{N}(\boldsymbol{\beta}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$, with a diagonal or full covariance matrix. For SIVI, we treat $\boldsymbol{\Sigma}$, diagonal or full, as a variational parameter and mix $\boldsymbol{\mu}$ with an MLP based implicit distribution. We consider three datasets: *waveform*, *spam*, and *nodal*. The details on datasets and inference are deferred to Appendix A.

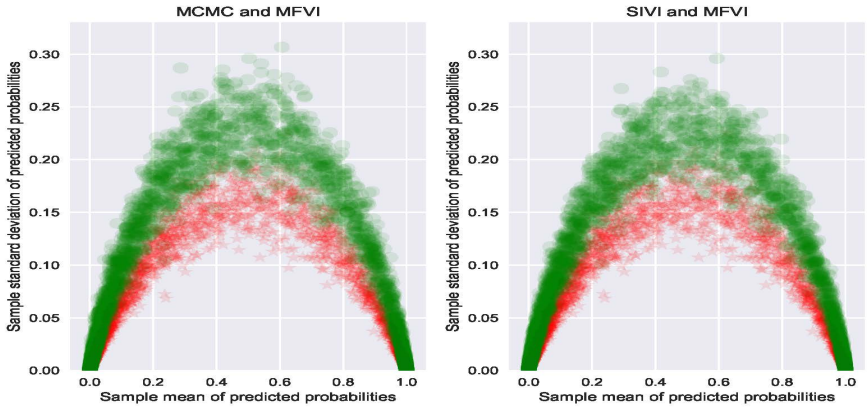


Figure 2.6: Comparison of MFVI (red), MCMC (green on left), and SIVI (green on right) with a full covariance matrix on quantifying predictive uncertainty for Bayesian logistic regression on *waveform*.

We collect $\boldsymbol{\beta}_j$ for $j = 1, \ldots, 1000$ to represent the inferred posterior $p(\boldsymbol{\beta} \,|\, \{\boldsymbol{x}_i, y_i\}_{1,N})$. For each test data $\boldsymbol{x}_{N+i}$, we calculate the predictive probabilities $1/(1 + e^{-\boldsymbol{x}_{N+i}^T \boldsymbol{\beta}_j})$ for all j and compute its sample mean, and sample standard deviation that measures the uncertainty of the predictive distribu-

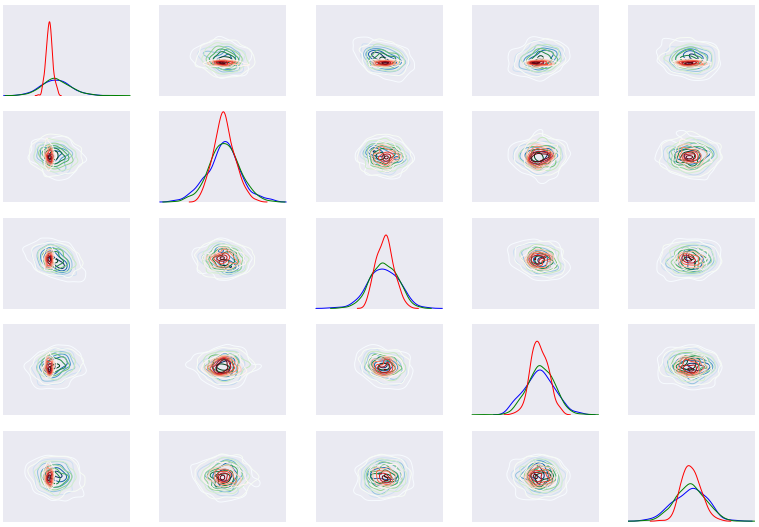


Figure 2.7: Marginal and pairwise joint posteriors for $(\beta_0, \ldots, \beta_4)$ inferred by MFVI (red), MCMC (blue), and SIVI (green, full covariance matrix) on *waveform*.

tion $p(y_{N+i} = 1 \,|\, \boldsymbol{x}_{N+i}, \{\boldsymbol{x}_i, y_i\}_{1,N})$. As shown in Figure 2.6, even with a full covariance matrix, the MVN variational distribution inferred by MFVI clearly underestimates the uncertainty in out-of-sample prediction, let alone with a diagonal one, whereas SIVI, mixing the MVN with an MLP based implicit distribution, closely matches MCMC in uncertainty estimation. As shown in Figure 2.7, the underestimation of predictive uncertainty by MFVI can be attributed to variance underestimation for both univariate marginal and pairwise joint posteriors, which are, by contrast, well agreed on between SIVI and MCMC.

Further examining all the univariate marginals, shown in Figure 2.9, correlation coefficients of $\boldsymbol{\beta}$, shown in Figure 2.8, and additional results, show

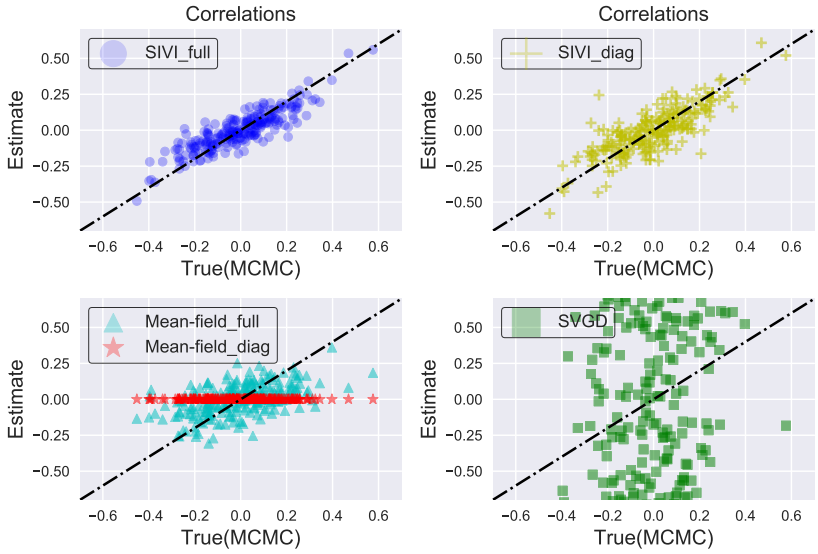


Figure 2.8: Correlation coefficients of $\boldsymbol{\beta}$ estimated from the posterior samples $\{\boldsymbol{\beta}_i\}_{i=1:1000}$ on *waveform*, compared with MCMC results. The closer to the dashed line the better.

in Appendix A, it is clear that SIVI well characterizes the posterior distribution of $\boldsymbol{\beta}$ and is only slightly negatively affected if its explicit layer is restricted with a diagonal covariance matrix, whereas MFVI with a diagonal/full covariance matrix and SVGD all clearly misrepresent the variance. Note we have also tried modified the code of variational boosting [89] for Bayesian logistic regression, but failed to obtain satisfactory results. We attribute the success of SIVI to its ability in better capturing the dependencies between β_v and supporting a highly expressive non-Gaussian variational distribution by mixing a MVN with a flexible implicit distribution, whose parameters can be efficiently optimized via an asymptotically exact surrogate ELBO.

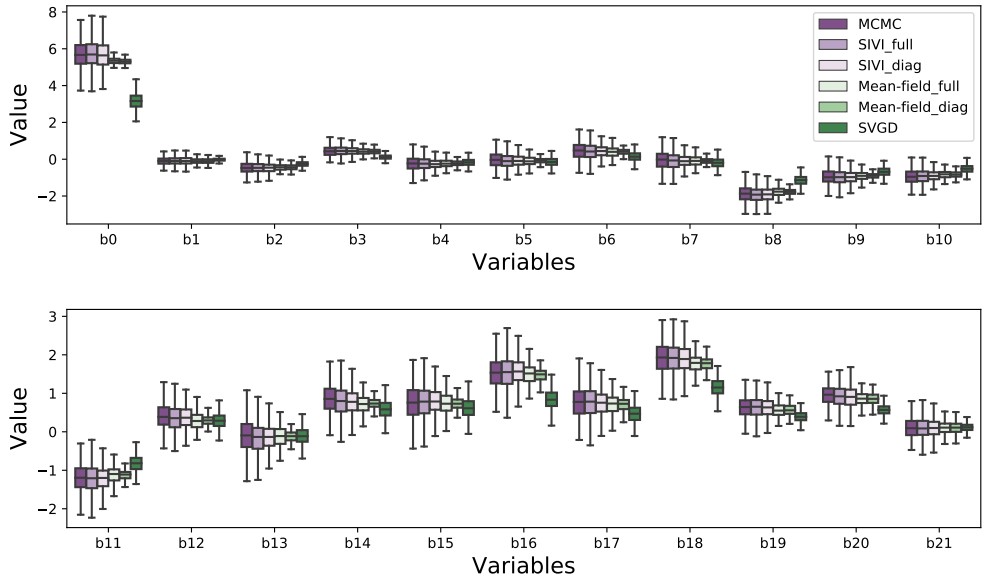


Figure 2.9: Comparison of all marginal posteriors of β_v inferred by various methods for Bayesian logistic regression on *waveform*.

2.4.4 Semi-Implicit Variational Autoencoder

Variational Autoencoder (VAE) [69, 117] is a popular generative model based approach for unsupervised feature learning and amortized inference. VAE iteratively infers the encoder parameter $\boldsymbol{\phi}$ and decoder parameter $\boldsymbol{\theta}$ to maximize the ELBO as

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z} \,|\, \boldsymbol{x})}[\log(p_{\boldsymbol{\theta}}(\boldsymbol{x} \,|\, \boldsymbol{z}))] - \mathcal{D}_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z} \,|\, \boldsymbol{x}) || p(\boldsymbol{z})).$$

The encoder distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z} \,|\, \boldsymbol{x})$ is required to be reparameterizable and analytically evaluable, which usually restricts it to a small exponential family. In particular, a canonical encoder is $q_{\boldsymbol{\phi}}(\boldsymbol{z} \,|\, \boldsymbol{x}) = \mathcal{N}(\boldsymbol{z} \,|\, \boldsymbol{\mu}(\boldsymbol{x}, \boldsymbol{\phi}), \boldsymbol{\Sigma}(\boldsymbol{x}, \boldsymbol{\phi}))$, where the Gaussian parameters are deterministically transformed from the observations

$\boldsymbol{x}$, via non-probabilistic deep neural networks parameterized by $\boldsymbol{\phi}$. Thus, given observation $\boldsymbol{x}_i$, its corresponding code $\boldsymbol{z}_i$ is forced to follow a Gaussian distribution, no matter how powerful the deep neural networks are. The Gaussian assumption, however, is often too restrictive to model skewness, kurtosis, and multimodality.

To this end, rather than using a single-stochastic-layer encoder, we use SIVI that can add multiple stochastic layers, as long as the first stochastic layer $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})$ remains to be reparameterizable and have an analytic PDF, and the layers added after are reparameterizable and simple to sample from. More specifically, we construct semi-implicit VAE (SIVAE) by using a hierarchical encoder that injects random noise at M different stochastic layers as

$$\begin{aligned}
&\boldsymbol{\ell}_t = T_t(\boldsymbol{\ell}_{t-1}, \boldsymbol{\epsilon}_t, \boldsymbol{x}; \boldsymbol{\phi}),\ \boldsymbol{\epsilon}_t \sim q_t(\boldsymbol{\epsilon}),\ t = 1, \ldots, M, \\
&\boldsymbol{\mu}(\boldsymbol{x}, \boldsymbol{\phi}) = f(\boldsymbol{\ell}_M, \boldsymbol{x}; \boldsymbol{\phi}), \quad \boldsymbol{\Sigma}(\boldsymbol{x}, \boldsymbol{\phi}) = g(\boldsymbol{\ell}_M, \boldsymbol{x}; \boldsymbol{\phi}), \\
&q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{x}, \boldsymbol{\phi}), \boldsymbol{\Sigma}(\boldsymbol{x}, \boldsymbol{\phi})),
\end{aligned} \tag{2.8}$$

where $\boldsymbol{\ell}_0 = \emptyset$ and T_t, f, and g are all deterministic neural networks. Note given data $\boldsymbol{x}_i$, $\boldsymbol{\mu}(\boldsymbol{x}_i, \boldsymbol{\phi})$, $\boldsymbol{\Sigma}(\boldsymbol{x}_i, \boldsymbol{\phi})$ are now random variables rather than following vanilla VAE to assume deterministic values. This clearly moves the encoder variational distribution beyond a simple Gaussian form.

To benchmark the performance of SIVAE, we consider the MNIST dataset that is stochastically banarized as in Salakhutdinov and Murray [121]. We use 55,000 for training and use the 10,000 observations in the testing set for performance evaluation. Similar to existing VAEs, we choose Bernoulli units,

Table 2.2: Comparison of the negative log evidence between various algorithms.

Methods	$-\log p(\boldsymbol{x})$
Results below form Burda et al. [19]	
VAE + IWAE	$= 86.76$
IWAE + IWAE	$= 84.78$
Results below form Salimans et al. [123]	
DLGM + HVI (1 leapfrog step)	$= 88.08$
DLGM + HVI (4 leapfrog step)	$= 86.40$
DLGM + HVI (8 leapfrog steps)	$= 85.51$
Results below form Rezende and Mohamed [116]	
DLGM+NICE [27] (k = 80)	≤ 87.2
DLGM+NF (k = 80)	≤ 85.1
Results below form Maaløe et al. [82]	
Auxiliary VAE (L=1, IW=1)	≤ 84.59
Results below form Mescheder et al. [88]	
VAE + IAF [70]	$\approx 84.9 \pm 0.3$
AVB + AC	$\approx 83.7 \pm 0.3$
SIVI (3 stochastic layers)	$= 84.07$
SIVI (3 stochastic layers)+ IW($\tilde{K} = 10$)	$= 83.25$

linked to a fully-connected neural network with two 500-unit hidden layers, as the decoder. Distinct from existing VAEs, whose encoders are often restricted to have a single stochastic layer, SIVI allows SIVAE to use MVN as its first stochastic layer, and draw the parameters of the MVN from $M = 3$ stochastic layers, whose structure is described in detail in Appendix A. As shown in Table 2.2 SIVAE achieves a negative log evidence of 84.07, which is further reduced to 83.25 if choosing importance reweighing with $\tilde{K} = 10$. In comparison to

other VAEs with a comparable single-stochastic-layer decoder, SIVAE achieves state-of-the-art performance by mixing an MVN with an implicit distribution defined as in (2.8) to construct a flexible encoder, whose marginal variational distribution is no longer restricted to the MVN distribution. We leave it for future study on further improving SIVAE by replacing the encoder MVN explicit layer with a normalizing flow, and adding convolution/autoregression to enrich the encoder's implicit distribution and/or the decoder.

2.5 Concluding Remarks

Combining the advantages of having analytic point-wise evaluable density ratios and tractable computation via Monte Carlo estimation, semi-implicit variational inference (SIVI) is proposed either as a black-box inference procedure, or to enrich mean-field variational inference with a flexible (implicit) mixing distribution. By designing a surrogate evidence lower bound that is asymptotically exact, SIVI establishes an optimization problem amenable to gradient ascend, without compromising the expressiveness of its semi-implicit variational distribution. Flexible but simple to optimize, SIVI approaches the accuracy of MCMC in quantifying posterior uncertainty in a wide variety of inference tasks, and is not constrained by conjugacy, often runs faster, and can generate independent posterior samples on the fly via the inferred stochastic variational inference network. The semi-implicit distribution can be applied as a generative model [163].

Chapter 3

Structured Variational Inference for Community Detection

This chapter, based on the publication [166], studies the convergence properties of structured variational inference in Stochastic Blockmodel. Mean-field variational inference (MFVI) has been widely applied in large scale Bayesian inference. However, the independence assumption of MFVI often leads to objective functions with many local optima, making optimization algorithms sensitive to initialization. In this chapter, we study the advantage of structured VI in the context of a simple two-class Stochastic Blockmodel. To facilitate theoretical analysis, the variational distribution is constructed to have a simple pairwise dependency structure on the nodes of the network. We prove that, in a broad density regime and for general random initializations, unlike MFVI, the estimated class labels by structured VI converge to the ground truth with high probability, when the model parameters are known, estimated within

The content in this chapter was published in [166], Yin, Mingzhang, YX Rachel Wang and Purnamrita Sarkar. "A Theoretical Case Study of Structured Variational Inference for Community Detection". In International Conference on Artificial Intelligence and Statistics. 2020. I mostly proposed the problem, designed the algorithm and implemented the methodology. All authors worked together in the theory proof, manuscript writing and revision. Prof. Wang and Prof. Sarkar helped in finalizing the experimental setting.

a reasonable range or jointly optimized with the variational parameters. In addition, empirically we demonstrate structured VI is more robust compared with MFVI when the graph is sparse and the signal to noise ratio is low. Our analysis takes a first step towards quantifying the role of added dependency structure in variational inference for community detection.

3.1 Theoretical Analyisis for Variational Inference

Variational inference (VI) is a widely used technique for approximating complex likelihood functions in Bayesian inference [65, 15, 62], and is known for its computational scalability. Nevertheless, theoretical understanding of its convergence properties is still an open area of research. Theoretical studies of variational methods (and similar algorithms that involve iteratively maximizing a lower bound) have drawn significant attention recently (see [9, 158, 159, 160, 75] for convergence properties of EM). For VI, the global optimizer of the variational lower bound is shown to be asymptotically consistent for a number of models including Latent Dirichlet Allocation (LDA) [15] and Gaussian mixture models [104]. In [153] the connection between VI estimates and profile M-estimation is explored and asymptotic consistency is established. In practice, however, it is well known the algorithm is not guaranteed to reach the global optimum and the performance of VI often suffers from local optima [16]. While in some models, convergence to the global optimum can be achieved with appropriate initialization [152, 8], understanding convergence with general initialization and the influence of local optima is less studied with a few

exceptions [158, 38, 95].

Mean-field variational inference (MFVI) has been widely used in probabilistic models. Despite being computationally scalable, MFVI suffers from many stability issues including symmetry-breaking, multiple local optima, and sensitivity to initialization, which are consequences of the non-convexity of typical mean-field problems [151, 61]. The independence assumption on latent variables also leads to the underestimation of posterior uncertainty [16, 161]. To address these problems, many studies suggest that modeling the latent dependency structure can expand the variational family under consideration and lead to larger ELBO and more stable convergence [157, 56, 39, 145, 113, 116, 161, 135]. However, rigorous theoretical analysis with convergence guarantees in this setting remains largely underexplored.

In this chapter, we aim to study the effect of added dependency structure in a MFVI framework. Since the behavior of the log-likelihood of MFVI is well understood for the very simple two class, equal sized Stochastic Blockmodel (SBM) [95, 168], we propose to add a simple pairwise link structure to MFVI in the context of inference for SBMs. We study how added dependency structure can improve MFVI. In particular, we focus on how random initialization behave for VI with added structure.

The stochastic blockmodel (SBM) [58] is a widely used network model for community detection in networks. There are a plethora of algorithms with theoretical guarantees for estimation for SBMs like Spectral methods [118, 22], semidefinite relaxation based methods [48, 108, 6], likelihood-based methods [5],

modularity based methods [131, 99, 11]. Among these, likelihood-based methods remain important and relevant due to their flexibility in incorporating additional model structures. Examples include mixed membership SBM [3, 169], networks with node covariates [115], dynamic networks [85], crowdsourced clustering [81]. Among likelihood based methods, VI provides a tractable approximation to the log-likelihood and is a scalable alternative to more expensive methods like Profile Likelihood [11], or MCMC based methods [131, 99]. Computationally, VI was also shown to scale up well to very large graphs [42].

On the theoretical front, [12] proved that the global optimum of MFVI behaves optimally in the dense degree regime. In terms of algorithm convergence, [168] showed the batch coordinate ascent algorithm (BCAVI) for optimizing the mean-field objective has guaranteed convergence if the initialization is sufficiently close to the ground truth. [95] fully characterized the optimization landscape and convergence regions of BCAVI for a simple two-class SBM with random initializations. It is shown that uninformative initializations can indeed converge to suboptimal local optima, demonstrating the limitations of the MFVI objective function.

Coming back to structured variational inference, it is important to note that, if one added dependencies between the posterior of each node, the natural approximate inference method is the belief propagation (BP) algorithm [105, 106, 154]. Based on empirical evidence, it has been conjectured in [24] that BP is asymptotically optimal for a simple two-class SBM. In the sparse setting where phase transition occurs, [94] analyzed a local variant of BP and

showed it is optimal given a specific initialization. In other parameter regions, rigorous theoretical understanding of BP, in particular, how adding dependence structure can improve convergence with general initializations is still an open problem.

Motivated by the above observations, we present a theoretical case study of structured variational inference for SBM. We emphasize here that our primary contribution *does not* lie in proposing a new estimation algorithm that outperforms state-of-the-art methods; rather we use this algorithm as an example to understand the interplay between a non-convex objective function and an iterative optimization algorithm with respect to random initializations, and compare it with MFVI. We consider a two-class SBM with equal class size, an assumption commonly used in theoretical work [94, 95] where the analysis for the simplest case is nontrivial.

We study structured VI by introducing a simple pairwise dependence structure between randomly paired nodes. By carefully bounding the mean field parameters and their logits in each iteration using a combination of concentration and Littlewood-Offord type anti-concentration arguments [29], we prove that in a broad density regime and under a fairly general random initialization scheme, the Variational Inference algorithm with Pairwise Structure (VIPS) can converge to the ground truth with probability tending to one, when the parameters are known, estimated within a reasonable range, or updated appropriately (Section 3.3). This is in contrast to MFVI, where convergence only happens for a narrower range of initializations. In addition, VIPS can

escape from certain local optima that exist in the MFVI objective. These results highlight the theoretical advantage of the added dependence structure. Empirically, we demonstrate that VIPS is more robust compared to MFVI when the graph is sparse and the signal to noise ratio is low (Section 3.4). We hope that our analysis can shed light on theoretical analysis of algorithms with more general dependence structure.

3.2 Problem Setup and Proposed Work

3.2.1 Preliminaries

The stochastic block model (SBM) is a generative network model with community structure. A K-community SBM for n nodes is generated as follows: each node is assigned to one of the communities in $\{1, \ldots, K\}$ according to a Multinomial distribution with parameter π. These memberships are represented by $U \in \{0,1\}^{n \times K}$, where each row follows an independent Multinomial $(1; \pi)$ distribution. We have $U_{ik} = 1$ if node i belongs to community k and $\sum_{k=1}^{K} U_{ik} = 1$. Given the community memberships, links between pairs of nodes are generated according to the entries in a $K \times K$ connectivity matrix B. That is, if A denotes the $n \times n$ binary symmetric adjacency matrix, then, for $i \neq j$,

$$P(A_{ij} = 1 | U_{ik} = 1, U_{j\ell} = 1) = B_{k\ell}. \tag{3.1}$$

We consider undirected networks, where both B and A are symmetric. Given an observed A, the goal is to infer the latent community labels U and the model parameters (π, B). Since the data likelihood $P(A; B, \pi)$ requires summing over

K^n possible labels, approximations such as MFVI are often needed to produce computationally tractable algorithms.

Throughout the rest of the chaper, we will use $\mathbf{1}_n$ to denote the all-one vector of length n. When it is clear from the context, we will drop the subscript n. Let I be the identity matrix and $J = \mathbf{1}\mathbf{1}^T$. $\mathbf{1}_C$ denotes a vector where the i-th element is 1 if $i \in C$ and 0 otherwise, where C is some index set. Similar to [95], we consider a two-class SBM with equal class size, where $K = 2$, $\pi = 1/2$, and B takes the form $B_{11} = B_{22} = p$, $B_{12} = B_{21} = q$, with $p > q$. We denote the two true underlying communities by G_1 and G_2, where G_1, G_2 form a partition of $\{1, 2, \ldots, n\}$ and $|G_1| = |G_2|$. (For convenience, we assume n is even.) As will become clear, the full analysis of structured VI in this simple case is highly nontrivial.

3.2.2 Variational Inference with Pairwise Structure (VIPS)

The well-known MFVI approximates the likelihood by assuming a product distribution over the latent variables. In other words, the posterior label distribution of the nodes is assumed to be independent in the variational distribution. To investigate how introducing dependence structure can help with the inference, we focus on a simple setting of linked pairs which are independent of each other. To be concrete, we randomly partition the n nodes into two sets: $P_1 = \{z_1, \cdots, z_m\}$, $P_2 = \{y_1, \cdots, y_m\}$, with $m = n/2$. Here $z_k, y_k \in \{1, \ldots, n\}$ are the node indices. In our structured variational distribution, we label pairs of nodes (z_k, y_k) using index $k \in \{1, \ldots, m\}$ and

assume there is dependence within each pair. The corresponding membership matrices for P_1 and P_2 are denoted by Z and Y respectively, which are both $m \times 2$ sub-matrices of the full membership matrix U. More explicitly, the k^{th} row of matrix Z encodes the membership of node z_k in P_1, and similarly for Y. For convenience, we permute both the rows and columns of A based on the node ordering in P_1 followed by that in P_2 to create a partitioned matrix: $A = \left[\begin{array}{c|c} A^{zz} & A^{zy} \\ \hline A^{yz} & A^{yy} \end{array}\right]$, where each block is an $m \times m$ matrix. Given the latent membership variable (Z, Y), by Eq. (3.1) the likelihood of A is given by

$$\begin{aligned} P(A^{zz}_{ij}|Z, B) &= \textstyle\prod_{a,b}[B_{ab}^{A^{zz}_{ij}}(1 - B_{ab})^{1-A^{zz}_{ij}}]^{Z_{ia}Z_{jb}} \\ P(A^{zy}_{ij}|Y, Z, B) &= \textstyle\prod_{a,b}[B_{ab}^{A^{zy}_{ij}}(1 - B_{ab})^{1-A^{zy}_{ij}}]^{Z_{ia}Y_{jb}} \\ P(A^{yy}_{ij}|Y, B) &= \textstyle\prod_{a,b}[B_{ab}^{A^{yy}_{ij}}(1 - B_{ab})^{1-A^{yy}_{ij}}]^{Y_{ia}Y_{jb}} \end{aligned} \tag{3.2}$$

where $a, b \in \{1, 2\}$ and $A^{zy} = (A^{yz})^T$.

A simple illustration of the partition and how ordered pairs of nodes are linked to incorporate dependence is given in Figure 3.1, where the the true underlying communities G_1 and G_2 are shaded differently. After the partition, we have m pairs of linked nodes indexed from 1 to m. For convenience of analysis, we define the following sets for these pairs of linked nodes, as illustrated in Figure 3.1.

Define C_1, (C_1') as the set of indices i of pairs (z_i, y_i) with $z_i \in G_1$, $(y_i \in G_1)$. Similarly, C_2, (C_2') is the set of indices of pairs (z_i, y_i) with $z_i \in G_2$, $(y_i \in G_2)$. We will also make use of the sets $C_{ab} \coloneqq C_a \cap C_b'$, where $a, b \in \{1, 2\}$. In Figure 3.1, as an illustrative example, the shaded nodes belong to community

G_1 and unshaded nodes belong to community G_2. The nodes are randomly partitioned into two sets P_1 and P_2, and pairs of nodes are linked from index 1 to m. Dependence structure within each linked pair is incorporated into the variational distribution $Q(Z, Y)$. For this partition and pair linking, $C_1 = \{4, 5\}$, $C_2 = \{1, 2, 3\}$, $C_1' = \{1, 2, 4\}$, $C_2' = \{3, 5\}$; $C_{11} = \{4\}$, $C_{12} = \{5\}$, $C_{21} = \{1, 2\}$, $C_{22} = \{3\}$.

We define the variational distribution for the latent membership matrix (Z, Y) as $Q(Z, Y)$, which we assume takes the form

$$Q(Z, Y) = \prod_{i=1}^{m} Q(Z_i, Y_i), \tag{3.3}$$

where Z_i denotes the i^{th} row of Z, and $Q(Z_i, Y_i)$ is a general categorical distribution with variational parameters defined as follows.

$$\psi_i^{cd} \coloneqq Q(Z_{i,c+1} = 1, Y_{i,d+1} = 1),$$

for $i \in \{1, \ldots, m\}, c, d \in \{0, 1\}$. This allows encoding more dependence structure between the posteriors at different nodes than vanilla MFVI, since we allow for dependence within each linked pair of nodes while keeping independence between different pairs. We define the marginal probabilities as:

$$\phi_i \coloneqq Q(Z_{i1} = 1) = \psi_i^{10} + \psi_i^{11}, \quad \xi_i \coloneqq Q(Y_{i1} = 1) = \psi_i^{01} + \psi_i^{11}. \tag{3.4}$$

Next we derive the ELBO on the data log-likelihood $\log P(A)$ using $Q(Z, Y)$. For pairwise structured variational inference (VIPS), ELBO takes the form

$$\mathcal{L}(Q; \pi, B) = \mathbb{E}_{Z,Y \sim Q(Z,Y)} \log P(A|Z, Y) - \mathcal{D}_{\mathrm{KL}}(Q(Z, Y)||P(Z, Y)),$$

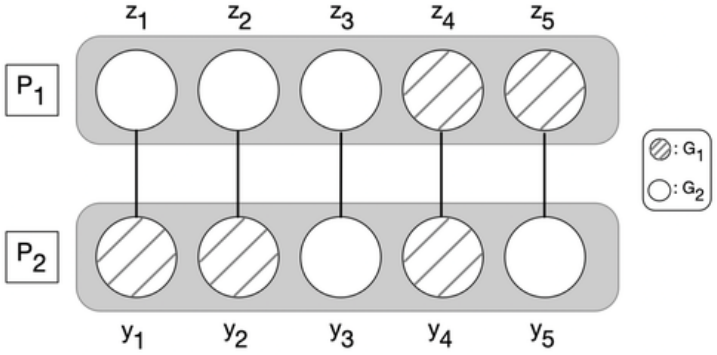


Figure 3.1: An illustration of a random pairwise partition, $n = 10$.

where $P(Z, Y)$ is the probability of community labels from SBM and follows independent Bernoulli (π) distribution, $\mathcal{D}_{\mathrm{KL}}(\cdot||\cdot)$ denotes the usual Kullback–Leibler divergence between two distributions. Using the likelihood in Eq. (3.2), the ELBO becomes

$$\begin{aligned}\mathcal{L}(Q;\pi,B) =& \frac{1}{2}\mathbb{E}_Q \sum_{i\neq j,a,b} [Z_{ia}Z_{jb}(A_{ij}^{zz}\alpha_{ab} + f(\alpha_{ab})) + Y_{ia}Y_{jb}(A_{ij}^{yy}\alpha_{ab} + f(\alpha_{ab}))] \\ &+\mathbb{E}_Q[\sum_{i\neq j,a,b} Z_{ia}Y_{jb}(A_{ij}^{zy}\alpha_{ab} + f(\alpha_{ab})) + \sum_{i,a,b} Z_{ia}Y_{ib}(A_{ii}^{zy}\alpha_{ab} + f(\alpha_{ab}))] \\ &-\sum_{i=1}^{m} \mathcal{D}_{\mathrm{KL}}(Q(z_i,y_i)||P(z_i)P(y_i)),\end{aligned} \tag{3.5}$$

where $\alpha_{ab} = \log(B_{ab}/(1-B_{ab}))$ and $f(\alpha) = -\log(1+e^{\alpha})$. The KL regularization term can be computed as

$$\mathcal{D}_{\mathrm{KL}}(Q(z_i,y_i)||P(z_i)P(y_i)) = \sum_{0\leq c,d\leq 1} \psi_i^{cd}\log(\psi_i^{cd})/(\pi^c\pi^d(1-\pi)^{1-c}(1-\pi)^{1-d}). \tag{3.6}$$

Our goal is to maximize $\mathcal{L}(Q;\pi,B)$ with respect to the variational parameters ψ_i^{cd} for $1 \leq i \leq m$. Since $\sum_{c,d}\psi_i^{cd} = 1$ for each i, it suffices to consider

ψ_i^{10}, ψ_i^{01} and ψ_i^{11}. By taking derivatives, we can derive a batch coordinate ascent algorithm for updating $\psi^{cd} = (\psi_1^{cd}, \ldots, \psi_m^{cd})$. Detailed calculation of the derivatives can be found in Section B.1 of the Appendix B. Recall that $\pi = \frac{1}{2}$. Also, define

$$t \coloneqq \frac{1}{2}\log\frac{p/(1-p)}{q/(1-q)}\ \lambda \coloneqq \frac{1}{2t}\log\frac{1-q}{1-p},\ \theta^{cd} \coloneqq \log\frac{\psi^{cd}}{1-\psi^{01}-\psi^{10}-\psi^{11}}, \quad (3.7)$$

where θ^{cd} are logits, $c, d \in \{0, 1\}$ and all the operations are defined *element-wise*.

Given the model parameters p, q, the current values of ψ^{cd} and the marginals $\phi = \psi^{10} + \psi^{11}$, $\xi = \psi^{01} + \psi^{11}$ as defined in Eq. (3.4), the updates for θ^{cd} are given by:

$$\begin{aligned}
\theta^{10} =& 4t[A^{zz} - \lambda(J-I)](\phi - \tfrac{1}{2}\mathbf{1}_m) - 2t(\mathrm{diag}(A^{zy}) - \lambda I)\mathbf{1}_m \\
&+ 4t[A^{zy} - \lambda(J-I) - \mathrm{diag}(A^{zy})](\xi - \frac{1}{2}\mathbf{1}_m), \quad (3.8)\\
\theta^{01} =& 4t[A^{yy} - \lambda(J-I)](\xi - \tfrac{1}{2}\mathbf{1}_m) - 2t(\mathrm{diag}(A^{yz}) - \lambda I)\mathbf{1}_m \\
&+ 4t[A^{yz} - \lambda(J-I) - \mathrm{diag}(A^{yz})](\phi - \frac{1}{2}\mathbf{1}_m), \quad (3.9)\\
\theta^{11} =& 4t[A^{zz} - \lambda(J-I)](\phi - \tfrac{1}{2}\mathbf{1}_m) + 4t[A^{yy} - \lambda(J-I)](\xi - \tfrac{1}{2}\mathbf{1}_m) \\
&+ 4t[A^{zy} - \lambda(J-I) - \mathrm{diag}(A^{zy})](\xi - \frac{1}{2}\mathbf{1}_m) \\
&+ 4t[A^{yz} - \lambda(J-I) - \mathrm{diag}(A^{yz})](\phi - \frac{1}{2}\mathbf{1}_m). \quad (3.10)
\end{aligned}$$

Given θ^{cd}, we can update the current values of ψ^{cd} and the corresponding marginal probabilities ϕ, ξ using element-wise operations as follows:

$$\begin{aligned}
\psi^{cd} &= \frac{e^{\theta^{cd}}}{1 + e^{\theta^{01}} + e^{\theta^{11}} + e^{\theta^{10}}}, \quad u \coloneqq (\phi, \xi) \\
\phi &= \frac{e^{\theta^{10}} + e^{\theta^{11}}}{1 + e^{\theta^{10}} + e^{\theta^{01}} + e^{\theta^{11}}}, \quad \xi = \frac{e^{\theta^{01}} + e^{\theta^{11}}}{1 + e^{\theta^{10}} + e^{\theta^{01}} + e^{\theta^{11}}}, \quad (3.11)
\end{aligned}$$

where $(c, d) = (1, 0), (0, 1), (1, 1)$. The marginal probabilities are concatenated as $u = (\phi, \xi) \in [0, 1]^n$. Thus u can be interpreted as the estimated posterior membership probability of all the nodes.

Since θ^{cd} determines ψ^{cd} in the categorical distribution and u represents the corresponding marginals, one can think of θ^{cd} and u as the local and global parameters respectively. It has been empirically shown that the structured variational methods can achieve better convergence property by iteratively updating the local and global parameters [15, 57, 56]. In the same spirit, in the full optimization algorithm, we update the parameters θ^{cd} and u iteratively by (3.8)–(3.11), following the order

$$\theta^{10} \to u \to \theta^{01} \to u \to \theta^{11} \to u \to \theta^{10} \cdots . \tag{3.12}$$

We call a full update of all the parameters $\theta^{10}, \theta^{01}, \theta^{11}, u$ in (3.12) as one *meta iteration* which consists of three inner iterations of u updates. We use $u_j^{(k)}$ $(j = 1, 2, 3)$ to denote the update in the j-th iteration of the k-th meta iteration, and $u^{(0)}$ to denote the initialization. Algorithm 2 gives the full algorithm when the model parameters are known.

Algorithm 2 Variational Inference with Pairwise Structure (VIPS)

input : Adjacency matrix $A \in \{0,1\}^{n\times n}$, model parameter $p, q, \pi = 1/2$.
output : The estimated node membership vector u.

Initialize the elements of u i.i.d. from an arbitrary distribution f_μ defined on $[0,1]$ with mean μ. Initialize $\theta^{10} = \theta^{01} = \theta^{11} = \mathbf{0}$;
Randomly select $n/2$ nodes as P_1 and the other $n/2$ nodes as P_2;
while *not converged* **do**
  Update θ^{10} by (3.8).
  Update $u = (\phi, \xi)$ by (3.11)
  Update θ^{01} by (3.9).
  Update $u = (\phi, \xi)$ by (3.11)
  Update θ^{11} by (3.10).
  Update $u = (\phi, \xi)$ by (3.11)
end

Remark 1. *So far we have derived the updates and described the optimization algorithm when the true parameters p, q are known. When they are unknown, they can be updated jointly with the variational parameters after each meta iteration as*

$$\begin{aligned} p =& \frac{\begin{array}{c}(\mathbf{1}_n - u)^T A(\mathbf{1}_n - u) + u^T A u \\ + 2(\mathbf{1}_m - \psi^{10} - \psi^{01})^T diag(A^{zy})\mathbf{1}_m\end{array}}{\begin{array}{c}(\mathbf{1}_n - u)^T (J - I)(\mathbf{1}_n - u) \\ + u^T (J - I)u + 2(\mathbf{1}_m - \psi^{10} - \psi^{01})^T \mathbf{1}_m\end{array}} \\ q =& \frac{(\mathbf{1}_n - u)^T A u + (\psi^{10} + \psi^{01})^T diag(A^{zy})\mathbf{1}_m}{(\mathbf{1}_n - u)^T (J - I)u_n + (\psi^{10} + \psi^{01})^T \mathbf{1}_m} \end{aligned} \tag{3.13}$$

Although it is typical to update p, q and u jointly, as shown in [95], analyzing MFVI updates with known parameters can shed light on the convergence behavior of the algorithm. Initializing u randomly while jointly updating p, q always leads MFVI to an uninformative local optima. For this reason, in what follows we will analyze Algorithm 2 in the context of both fixed and updating p, q.

3.3 Main Results

In this section, we present theoretical analysis of the algorithm in three settings: (i) When the parameters are set to the true model parameters p, q; (ii) When the parameters are not too far from the true values, and are held fixed throughout the updates; (iii) Starting from some reasonable guesses of the parameters, they are jointly updated with latent membership estimates.

In the following analysis, we will frequently use the eigen-decomposition of the expected adjacency matrix $P = \mathbb{E}[A|U] = \frac{p+q}{2}\mathbf{1}_n\mathbf{1}_n^T + \frac{p-q}{2}v_2v_2^T - pI$ where $v_2 = (v_{21}, v_{22})^T = (\mathbf{1}_{C_1} - \mathbf{1}_{C_2}, \mathbf{1}_{C_1'} - \mathbf{1}_{C_2'})^T$ is the second eigenvector. Since the second eigenvector is just a shifted and scaled version of the membership vector, the projection $|\langle u, v_2\rangle|$ is equivalent to the ℓ_1 error from true label z^* (up-to label permutation) by $\|u - z^*\|_1 = m - |\langle u, v_2\rangle|$. We consider the parametrization $p \asymp q \asymp \rho_n$, where the density $\rho_n \to 0$ at some rate and $p - q = \Omega(\rho_n)$.

When the true parameters p, q are known, it has been shown [125] that without dependency structure, MFVI with random initializations converges to the stationary points with non-negligible probability. When the variational distribution has a simple pairwise dependency structure as VIPS, we show a stronger result. To be concrete, in this setting, we establish that convergence happens with probability approaching 1. In addition, unlike MFVI, the convergence holds for general random initializations. We will first consider the situation when $u^{(0)}$ is initialized from a distribution centered at $\mu = \frac{1}{2}$ and show the results for $\mu \neq \frac{1}{2}$ in Corollary 1.

Theorem 2 (Sample behavior for known parameters)**.** *Assume* $\theta^{10}, \theta^{01}, \theta^{11}$ *are initialized as* $\mathbf{0}$ *and the elements of* $u^{(0)} = (\phi^{(0)}, \xi^{(0)})$ *are initialized i.i.d. from Bernoulli*$(\frac{1}{2})$*. When* $p \asymp q \asymp \rho_n$*,* $p - q = \Omega(\rho_n)$*, and* $\sqrt{n}\rho_n = \Omega(\log(n))$*, Algorithm 2 converges to the true labels asymptotically after the second meta iteration, in the sense that*

$$\|u_3^{(2)} - z^*\|_1 = n \exp(-\Omega_P(n\rho_n)))$$

z^* *are the true labels with* $z^* = \mathbf{1}_{G_1}$ *or* $\mathbf{1}_{G_2}$*. The same convergence holds for all the later iterations.*

Proof. We provide a proof sketch here and defer the details to Section B.2 of the Appendix B. We assume for the first six iterations, we randomly partition A into six $A^{(i)}, i = 0, \ldots, 5$ by assigning each edge to one of the six subgraphs with equal probability. For the later iterations, we can use the whole graph A. Then $A^{(i)}$'s are independent with population matrix $P/6$. Although not used in Algorithm 2, the graph splitting is a widely used technique for theoretical convenience [87, 20] and allows us to bound the noise in each iteration more easily. The main arguments involve lower bounding the size of the projection $|\langle u, v_2 \rangle|$ in each iteration as it increases towards $n/2$, at which point the algorithm achieves strong consistency. For ease of exposition, we will scale everything by 6 so that p, q, λ correspond to the parameters for the full un-split matrix P. This does not affect the analysis in any way.

In each iteration, we decompose the intermediate $\theta^{10}, \theta^{01}, \theta^{11}$ into block-wise constant signal and random noise using the spectral property of the

population matrix P. As an illustration, in the first meta iteration, we write the update in (3.8)–(3.10) as signal plus noise,

$$\theta_i^{10} = 4t(s_1\mathbf{1}_{C_1} + s_2\mathbf{1}_{C_2} + r_i^{(0)}), \quad \theta_i^{01} = 4t(x_1\mathbf{1}_{C_1'} + x_2\mathbf{1}_{C_2'} + r_i^{(1)})$$
$$\theta_i^{11} = 4t(y_1\mathbf{1}_{C_1} + y_2\mathbf{1}_{C_2} + y_1\mathbf{1}_{C_1'} + y_2\mathbf{1}_{C_2'} + r_i^{(2)})$$

where t is a constant and the noise has the form

$$r^{(i)} = R^{(i)}(u_j^{(k)} - \frac{1}{2}\mathbf{1}) \tag{3.14}$$

for appropriate j, k, where $R^{(i)}$ arises from the sample noise in the adjacency matrix. We handle the noise from the first iteration $r^{(0)}$ with a Berry-Esseen bound conditional on $u^{(0)}$, and the later $r^{(i)}$ with a uniform bound. The blockwise constant signals s_1, x_1, y_1 are updated as $(\frac{p+q}{2} - \lambda)(\langle u, \mathbf{1}_n\rangle - m) + (\frac{p-q}{2})\langle u, v_2\rangle$ and s_2, x_2, y_2 are updated as $(\frac{p+q}{2} - \lambda)(\langle u, \mathbf{1}_n\rangle - m) - (\frac{p-q}{2})\langle u, v_2\rangle$. As $\langle u, v_2\rangle$ increases throughout the iterations, the signals become increasingly separated for the two communities. Using Littlewood-Offord type anti-concentration, we show in the first meta iteration,

$$\begin{aligned}
&\langle u_1^{(1)}, v_2\rangle = \Omega_P(n\sqrt{\rho_n}), \quad \langle u_1^{(1)}, \mathbf{1}\rangle - m = 0 \\
&\langle u_2^{(1)}, v_2\rangle \geq \frac{n}{8} - o_P(n), \quad \langle u_2^{(1)}, \mathbf{1}\rangle - m = 0, \quad \langle u_3^{(1)}, v_2\rangle \geq \frac{1}{4}n + o_P(n) \\
&-\frac{n}{8} - o_P(n) \leq \langle u_3^{(1)}, \mathbf{1}\rangle - m \leq \frac{n}{4} + o_P(n)
\end{aligned} \tag{3.15}$$

After the second meta iteration we have

$$\begin{aligned}
s_1^{(2)}, x_1^{(2)}, y_1^{(2)} = \Omega_P(n\rho_n), \quad s_2^{(2)}, x_2^{(2)}, y_2^{(2)} = -\Omega_P(n\rho_n) \\
2y_1^{(2)} - s_1^{(2)} = \Omega_P(n\rho_n), 2y_1^{(2)} - x_1^{(2)} = \Omega_P(n\rho_n); \\
s_1^{(2)} - (y_1^{(2)} + y_2^{(2)}) = \Omega_P(n\rho_n), \quad x_1^{(2)} - (y_1^{(2)} + y_2^{(2)}) = \Omega_P(n\rho_n);
\end{aligned} \tag{3.16}$$

Plugging (3.16) to (3.11), we have the desired convergence after the second meta iteration. □

The next corollary shows the same convergence holds when we use a general random initialization not centered at 1/2. In contrast, MFVI converges to stationary points $\mathbf{0}_n$ or $\mathbf{1}_n$ with such initializations.

Corollary 1. *Assume the elements of $u^{(0)}$ are i.i.d. sampled from a distribution with mean $\mu \neq 0.5$. Under the conditions in Theorem 2, applying Algorithm 2 with known p, q, we have $\|u_1^{(3)} - z^*\|_1 = n \exp(-\Omega_P(n\rho_n)))$. The same order holds for all the later iterations.*

The proof relies on showing after the first iteration, $u_1^{(1)}$ behaves like nearly independent Bernoulli($\frac{1}{2}$), the details of which can be found in Appendix B, sectionB.2.

The next proposition focuses on the behavior of special points in the optimization space for u. In particular, we show that Algorithm 2 enables us to move away from the stationary points $\mathbf{0}_n$ and $\mathbf{1}_n$, whereas in MFVI, the optimization algorithm gets trapped in these stationary points [95].

Proposition 4 (Escaping from stationary points)**.**

(i) $(\psi^{00}, \psi^{01}, \psi^{10}, \psi^{11}) = (\mathbf{1}_m, \mathbf{0}_m, \mathbf{0}_m, \mathbf{0}_m)$, $(\mathbf{0}_m, \mathbf{0}_m, \mathbf{0}_m, \mathbf{1}_m)$ *are the stationary points of the pairwise structured ELBO when p, q are known, which maps to $u = \mathbf{0}_n$ and $\mathbf{1}_n$ respectively.*

(ii) With the updates in Algorithm 2, when $u^{(0)} = \mathbf{0}_n$, $\mathbf{1}_n$, *VIPS converges to the true labels with* $\|u_1^{(3)} - z^*\|_1 = n \exp(-\Omega_P(n\rho_n)))$.

The above results requires knowing the true p and q. The next proposition shows that, even if we do not have access to the true parameters, as long as some reasonable estimates can be obtained, the same convergence as in Theorem 2 holds thus demonstrating robustness to misspecified parameters. Here we hold the parameters fixed and only update u as in Algorithm 2. When $\hat{p}, \hat{q} \asymp \rho_n$, we need $\hat{p} - \hat{q} = \Omega(\rho_n)$ and $\hat{p}, \hat{q}$ not too far from the true values to achieve convergence. The proof is deferred to the Appendix B.

Proposition 5 (Parameter robustness)**.** *If we replace true* p, q *with some estimation* $\hat{p}, \hat{q}$ *in Algorithm 2, the same conclusion as in Theorem 2 holds if*

$$1.\ \tfrac{p+q}{2} > \hat{\lambda}, \quad 2.\ \hat{\lambda} - q = \Omega(\rho_n), \quad 3.\ \hat{t} = \Omega(1).$$

where $\hat{t} = \frac{1}{2} \log \dfrac{\hat{p}/(1-\hat{p})}{\hat{q}/(1-\hat{q})}$, $\hat{\lambda} = \frac{1}{2\hat{t}} \log \dfrac{1-\hat{q}}{1-\hat{p}}$.

Finally, we consider updating the parameters jointly with u (as explained in Remark 1) by first initializing the algorithm with some reasonable $p^{(0)}, q^{(0)}$.

Theorem 3 (Updating parameters and u simultaneously)**.** *Suppose we initialize with some estimates of true* (p, q) *as* $\hat{p} = p^{(0)}$, $\hat{q} = q^{(0)}$ *satisfying the conditions in Proposition 5 and apply two meta iterations in Algorithm 2 to update* u *before updating* $\hat{p} = p^{(1)}, \hat{q} = q^{(1)}$. *After this, we alternate between updating* u

and the parameters after each meta iteration. Then

$$p^{(1)} = p + O_P(\sqrt{\rho_n}/n),\ q^{(1)} = q + O_P(\sqrt{\rho_n}/n),\ \|u_3^{(2)} - z^*\|_1 = n\exp(-\Omega(n\rho_n)),$$

and the same holds for all the later iterations.

3.4 Experimental Results

In this section, we present some numerical results. In Figures 3.2 to 3.4 we show the effectiveness of VIPS in our theoretical setting of two equal sized communities. In Figures 3.6 (a) and (b) we show that empirically the advantage of VIPS holds even for unbalanced community sizes and $K > 2$. Our goal is two-fold: (i) we demonstrate that the empirical convergence behavior of VIPS coincides well with our theoretical analysis in Section 3.3; (ii) in practice VIPS has superior performance over MFVI in both the simple setting we have analyzed and more general settings, thus confirming the advantage of the added dependence structure. For the sake of completeness, we also include comparisons with other popular algorithms, even though it is not our goal to show VIPS outperforms these methods.

In Figure 3.2, we compare the convergence property of VIPS with MFVI for initialization from independent Bernoulli's with means $\mu = 0.1, 0.5$, and 0.9. We randomly generate a graph with $n = 3000$ nodes with parameters $p_0 = 0.2, q_0 = 0.01$ and show results from 20 random trials. We plot $\min(\|u - z^*\|_1, \|u - (\mathbf{1} - z^*)\|_1)$, or the ℓ_1 distance of the estimated label u to the ground truth z^* on the Y axis versus the iteration number on the X axis. In this

experiments, both VIPS and MFVI were run with the true p_0, q_0 values. As shown in Figure 3.2, when $\mu = \frac{1}{2}$, VIPS converges to z^* after two meta iterations (6 iterations) for all the random initializations. In contrast, for MFVI, a fraction of the random initializations converge to $\mathbf{0}_n$ and $\mathbf{1}_n$. When $\mu \neq \frac{1}{2}$, VIPS converges to the ground truth after three meta iterations, whereas MFVI stays at the stationary points $\mathbf{0}_n$ and $\mathbf{1}_n$. This is consistent with our theoretical results in Theorem 2 and Corollary 1, and those in [95].

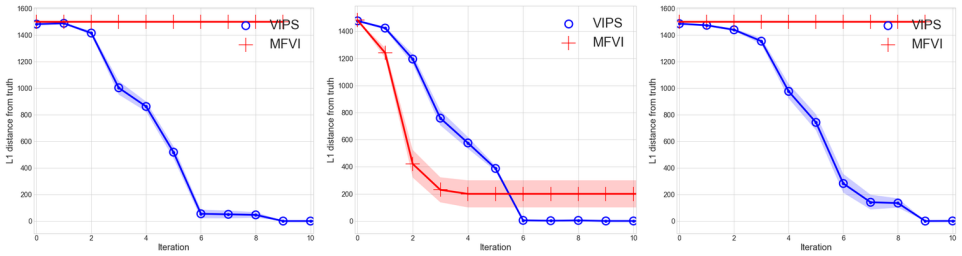


Figure 3.2: ℓ_1 distance from ground truth (Y axis) vs. number of iterations (X axis). The line is the mean of 20 random trials and the shaded area shows the standard deviation. u is initialized from i.i.d. Bernoulli with mean $\mu = 0.1, 0.5, 0.9$ from the left to right.

In Figure 3.3, we show when the true p, q are unknown, the dependence structure makes the algorithm more robust to estimation errors in $\hat{p}, \hat{q}$. The heatmap represents the normalized mutual information (NMI) [119] between u and z^*, with $\hat{p}$ on the X axis and $\hat{q}$ on the Y axis. We only examine pairs with $\hat{p} > \hat{q}$. Both VIPS and MFVI were run with $\hat{p}$ and $\hat{q}$, which were held fixed and differ from the true values to varying extent. The dashed line represents the true p, q used to generate the graph. For each $\hat{p}, \hat{q}$ pair, the mean NMI for 20 random initializations from i.i.d Bernoulli($\frac{1}{2}$) is shown. VIPS recovers the

ground truth in a wider range of $\hat{p}, \hat{q}$ values than MFVI.

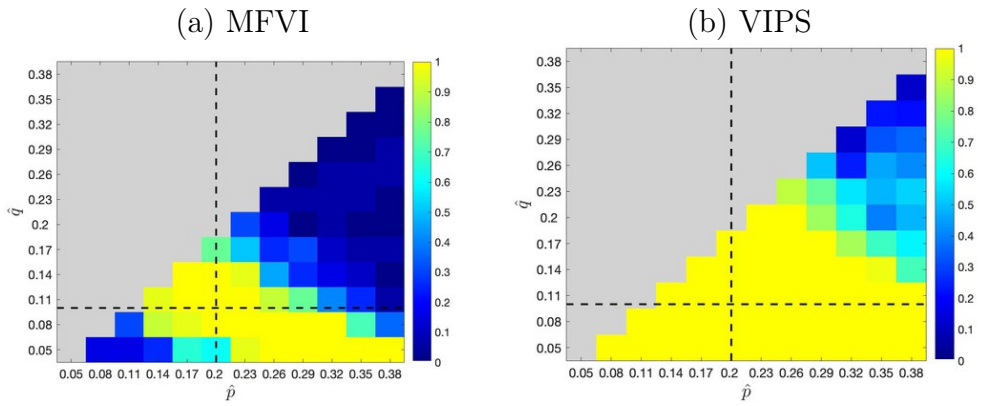


Figure 3.3: NMI averaged over 20 random initializations for each $\hat{p}$, $\hat{q}$ ($\hat{p} > \hat{q}$). The true parameters are $(p_0, q_0) = (0.2, 0.1)$, $\pi = 0.5$ and $n = 2000$. The dashed lines indicate the true parameter values.

In Figure 3.4, we compare VIPS with MFVI under different network sparsities and signal-to-noise ratios (SNR) as defined by $r_0 = p_0/q_0$. For the sake of completeness, we also include two other popular algorithms, Belief Propagation (BP) [25] and Spectral Clustering [118]. We plot the mean and standard deviation of NMI for 20 random trials in each setting. In each trial, to meet the conditions in Theorem 3, we started VIPS with $\hat{p}$ equal to the average degree of A, and $\hat{q} = \hat{p}/r_0$. $\hat{p}$ and $\hat{q}$ were updated alternatingly with u according to Eq. (3.13) after three meta iterations in Algorithm 2, a setting similar to that of Theorem 3.

In Figure 3.4-(a), the average expected degree is fixed at 70 as the SNR p_0/q_0 increases on the X axis, whereas in Figure 3.4-(b), the SNR is fixed at 2 and we vary the average expected degree on the X axis. The results show

that VIPS consistently outperforms MFVI, indicating the advantage of the added dependence structure. Note that we plot BP with the model parameters initialized at true (p_0, q_0) , since it is sensitive to initialization setting, and behaves poorly with mis-specified ones. Despite this, VIPS is largely comparable to BP and Spectral Clustering. For average degree 20 (Figure 3.4-(b)), BP outperforms all other methods, because of the correct parameter setting. This NMI value becomes 0.4 with high variance, if we provide initial $\hat{p}, \hat{q}$ values to match the average degree but $\hat{p}/\hat{q} = 10$. In contrast, VIPS is much more robust to the initial choice of $\hat{p}, \hat{q}$, which we show in Figure 3.5.

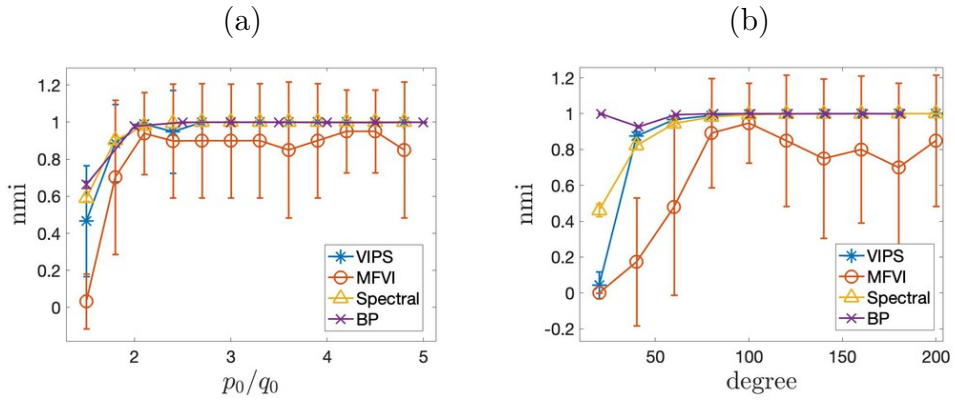


Figure 3.4: Comparison of NMI under different SNR p_0/q_0 and network degrees by means and standard deviations from 20 random trials, $n = 2000$.

We further show that VIPS with fixed mis-specified parameters (within reasonable deviation from the truth), fixed true parameters and parameters updated with Eq. (3.13) converge to the truth when initialized by independent Bernoulli's. In Figure 3.5, we compare VIPS and MFVI with and without parameter updates. In the first scheme, for VIPS, we do parameter updates

after 3rd meta iteration onward, and for fairness, we start parameter updates 9 iterations onward for MFVI. The other scheme has $\hat{p}, \hat{q}$ held fixed. In both schemes, the VIPS performs better than MFVI.

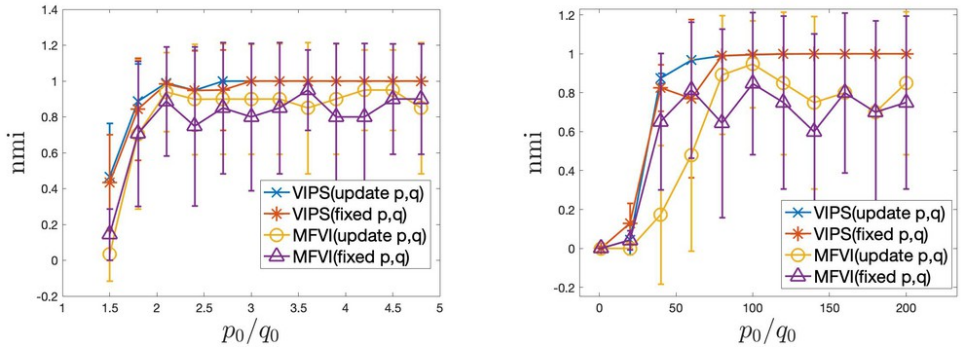


Figure 3.5: Two schemes for estimating model parameters for VIPS and MFVI. Both use the initial $\hat{p}$ and $\hat{q}$ as described in Figure 3.4.

3.5 Discussion and Generalizations

In this chaper, we propose a simple Variational Inference algorithm with Pairwise Structure (VIPS) in a SBM with two equal sized communities. VI has been extensively applied in the latent variable models mainly due to their scalability and flexibility for incorporating changes in model structure. However, theoretical understanding of the convergence properties is limited and mostly restricted to the mean field setting with fully factorized variational distributions (MFVI). Theoretically we prove that in a SBM with two equal sized communities, VIPS can converge to the ground truth with probability tending to one for different random initialization schemes and a range of

graph densities. In contrast, MFVI only converges for a constant fraction of Bernoulli(1/2) random initializations. We consider settings where the model parameters are known, estimated or appropriately updated as part of the iterative algorithm.

Though our main results are for $K = 2, \pi = 0.5$, we conclude with a discussion on generalizations to unbalanced clusters and SBMs with $K > 2$ equal communities. To apply VIPS for d $K > 2$ clusters, we will have $K^2 - 1$ categorical distribution parameters ψ^{cd} for $c, d \in \{1, 2, \ldots, K\}$ and marginal likelihood $\phi_1, \ldots, \phi_{K-1}$, $\xi_1, \ldots, \xi_{K-1}$. The updates are similar to Eq. (3.10) and Eq. (3.11). Similar to the $K = 2$ case, we update the local and global parameters iteratively. As for the unbalanced case, the updates involve an additional term which is the logit of π. We assume that π is known and fixed.

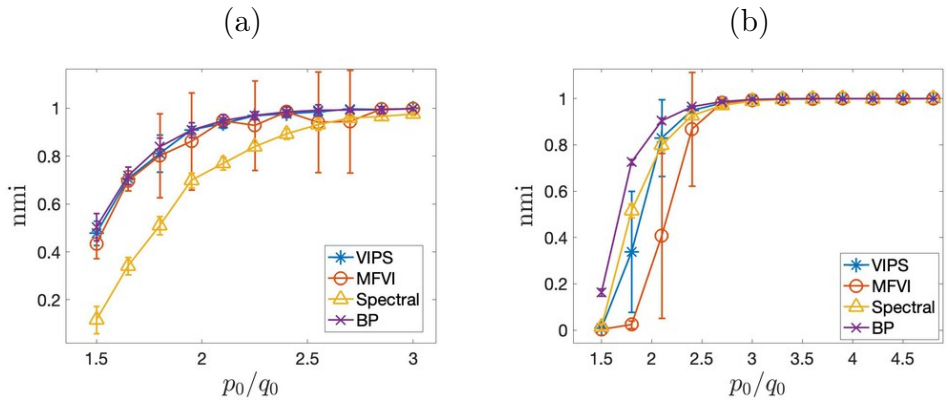


Figure 3.6: Comparison of VIPS, MFVI, Spectral and BP with 20 random trials for $n = 2000$, average degree 50, p_0/q_0 is changed on X axis. (a) $\pi = 0.3$ (b) K =3, $B = (p - q)I + qJ$.

In Figure 3.6-(a), we show results for unbalanced SBM with $\pi = 0.3$,

which is assumed to be known. In Figure 3.6-(b), similar to the setting in [95], we consider a SBM with three equal-sized communities. The parameters are set as $n = 2000$, average degree 50, p_0 and q_0 are changed to get different SNR values and the random initialization is from Dirichlet$(1, 1, 1)$. For a fair comparison of VIPS, MFVI and BP, we use the true p_0, q_0 values in all three algorithms; robustness to parameter specification of VIPS is shown in Figure 3.5. We see that for the unbalanced setting (Figure 3.6-(a)) VIPS performs as well as BP and better than Spectral Clustering. For the $K = 3$ setting (Figure 3.6-(b)) VIPS performs worse than BP and Spectral for very low SNR values, whereas for higher SNR it performs comparably to Spectral and BP, and better than MFVI, which has much higher variance.

Chapter 4

Variational Inference with Discrete Latent Variables

This chapter, based on two publications [162, 164], studies variance reduction for the variational inference with discrete latent variables. To estimate the gradient of variational parameters, we propose the augment-REINFORCE-merge (ARM) estimator that is unbiased, exhibits low variance, and has low computational complexity. Exploiting variable augmentation, REINFORCE, and reparameterization, the ARM estimator achieves adaptive variance reduction for Monte Carlo integration by merging two expectations via common random numbers. The variance-reduction mechanism of the ARM estimator can also be attributed to either antithetic sampling in an augmented space, or the use of an optimal anti-symmetric "self-control" baseline function. Experimental results show the ARM estimator provides superior performance in

The content in this chapter was published in [162], Yin, Mingzhang, Mingyuan Zhou. "ARM: Augment-REINFORCE-Merge Gradient for Stochastic Binary Networks". In International Conference on Learning Representations, 2019. I designed the methodology with Prof. Zhou, mostly did the theoretical analysis, implemented the methodology and wrote the draft paper. Prof. Zhou proposed the initial methodology, helped in the experimental design, the rewriting and revision.

auto-encoding variational inference and maximum likelihood estimation, for discrete latent variable models with one or multiple stochastic binary layers.

4.1 Optimization for Discrete Latent Variable Models

Given a function $f(\boldsymbol{z})$ of a random variable $\boldsymbol{z} = (z_1, \ldots, z_V)^T$, which follows a distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z})$ parameterized by $\boldsymbol{\phi}$, there has been significant recent interest in estimating $\boldsymbol{\phi}$ to maximize (or minimize) the expectation of $f(\boldsymbol{z})$ with respect to $\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z})$, expressed as

$$\mathcal{E}(\boldsymbol{\phi}) = \int f(\boldsymbol{z}) q_{\boldsymbol{\phi}}(\boldsymbol{z}) d\boldsymbol{z} = \mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z})}[f(\boldsymbol{z})]. \tag{4.1}$$

In particular, this expectation objective appears in both maximizing the evidence lower bound (ELBO) for variational inference [66, 16] and approximately maximizing the log marginal likelihood of a hierarchal Bayesian model [13], two fundamental problems in statistical inference.

To maximize (4.1), if $\nabla_{\boldsymbol{z}} f(\boldsymbol{z})$ is tractable to compute and $\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z})$ can be generated via reparameterization as $\boldsymbol{z} = \mathcal{T}_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}),\ \boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon})$, where $\boldsymbol{\epsilon}$ are random noises and $\mathcal{T}_{\boldsymbol{\phi}}(\cdot)$ denotes a deterministic transform parameterized by $\boldsymbol{\phi}$, then one may apply the reparameterization trick [69, 117] to compute the gradient as

$$\nabla_{\boldsymbol{\phi}} \mathcal{E}(\boldsymbol{\phi}) = \nabla_{\boldsymbol{\phi}} \mathbb{E}_{\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon})}[f(\mathcal{T}_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}))] = \mathbb{E}_{\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon})}[\nabla_{\boldsymbol{\phi}} f(\mathcal{T}_{\boldsymbol{\phi}}(\boldsymbol{\epsilon}))]. \tag{4.2}$$

This trick, however, is often inapplicable to discrete random variables, as widely used to construct discrete latent variable models such as sigmoid belief

networks [98, 128]. To maximize (4.1) for discrete $\boldsymbol{z}$, using the score function $\nabla_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\boldsymbol{z}) = \nabla_{\boldsymbol{\phi}} q_{\boldsymbol{\phi}}(\boldsymbol{z}) / q_{\boldsymbol{\phi}}(\boldsymbol{z})$, one may compute $\nabla_{\boldsymbol{\phi}} \mathcal{E}(\boldsymbol{\phi})$ via REINFORCE [155] as

$$\nabla_{\boldsymbol{\phi}} \mathcal{E}(\boldsymbol{\phi}) = \mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z})}[f(\boldsymbol{z}) \nabla_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\boldsymbol{z})] \approx \frac{1}{K} \sum\nolimits_{k=1}^{K} f(\boldsymbol{z}^{(k)}) \nabla_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\boldsymbol{z}^{(k)}),$$

where $\boldsymbol{z}^{(k)} \overset{iid}{\sim} q_{\boldsymbol{\phi}}(\boldsymbol{z})$ are independent, and identically distributed (iid). This unbiased estimator is also known as (a.k.a.) the score-function [34] or likelihood-ratio estimator [40]. While it is unbiased and only requires drawing iid random samples from $q_{\boldsymbol{\phi}}(\boldsymbol{z})$ and computing $\nabla_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\boldsymbol{z}^{(k)})$, its high Monte-Carlo-integration variance often limits its use in practice. Note that if $f(\boldsymbol{z})$ depends on $\boldsymbol{\phi}$, then we assume it is true that $\mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z})}[\nabla_{\boldsymbol{\phi}} f(\boldsymbol{z})] = 0$. For example, in variational inference, we need to maximize the ELBO as $\mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z})}[f(\boldsymbol{z})]$, where $f(\boldsymbol{z}) = \log[p(\boldsymbol{x} \,|\, \boldsymbol{z}) p(\boldsymbol{z}) / q_{\boldsymbol{\phi}}(\boldsymbol{z})]$. In this case, although $f(\boldsymbol{z})$ depends on $\boldsymbol{\phi}$, as $\mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z})}[\nabla_{\boldsymbol{\phi}} \log q_{\boldsymbol{\phi}}(\boldsymbol{z})] = \int \nabla_{\boldsymbol{\phi}} q_{\boldsymbol{\phi}}(\boldsymbol{z}) d\boldsymbol{z} = \nabla_{\boldsymbol{\phi}} \int q_{\boldsymbol{\phi}}(\boldsymbol{z}) d\boldsymbol{z} = 0$, we have $\mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z})}[\nabla_{\boldsymbol{\phi}} f(\boldsymbol{z})] = 0$.

To address the high-variance issue, one may introduce an appropriate baseline (a.k.a. control variate) to reduce the variance of REINFORCE [102, 112, 90, 47, 91, 73, 96, 51]. Alternatively, one may first relax the discrete random variables with continuous ones and then apply the reparameterization trick to estimate the gradients, which reduces the variance of Monte Carlo integration at the expense of introducing bias [84, 64]. Combining both REINFORCE and the continuous relaxation of discrete random variables, REBAR of Tucker et al. [147] and RELAX of Grathwohl et al. [45] both aim to produce a low-variance and unbiased gradient estimator by introducing a continuous relaxation

based baseline function, whose parameters, however, need to be estimated at each mini-batch by minimizing the sample variance of the estimator with stochastic gradient descent (SGD). Estimating the baseline parameters often clearly increases the computation. Moreover, the potential conflict, between *minimizing* the sample variance of the gradient estimate and *maximizing* the expectation objective, could slow down or even prevent convergence and increase the risk of overfitting. Another interesting variance-control idea applicable to discrete latent variables is using local expectation gradients, which estimates the gradients based on REINFORCE, by performing Monte Carlo integration using a single global sample together with exact integration of the local variable for each latent dimension [143].

Distinct from the usual idea of introducing baseline functions and optimizing their parameters to reduce the estimation variance of REINFORCE, we propose the augment-REINFORCE-merge (ARM) estimator, a novel unbiased and low-variance gradient estimator for binary latent variables that is also simple to implement and has low computational complexity. We show by rewriting the expectation with respect to Bernoulli random variables as one with respect to augmented exponential random variables, and then expressing the gradient as an expectation via REINFORCE, one can derive the ARM estimator in the augmented space with the assistance of appropriate reparameterization. In particular, in the augmented space, one can derive the ARM estimator by using either the strategy of sharing common random numbers between two expectations, or the strategy of applying antithetic sampling. Both

strategies, as detailedly discussed in Owen [101], can be used to explain why the ARM estimator is unbiased and could lead to significant variance reduction. Moreover, we show that the ARM estimator can be considered as improving the REINFORCE estimator in an augmented space by introducing an optimal baseline function subject to an anti-symmetric constraint; this baseline function can be considered as a "self-control" one, as it exploits the function f itself and correlated random noises for variance reduction, and adds no extra parameters to learn. This "self-control" feature makes the ARM estimator distinct from both REBAR and RELAX, which rely on minimizing the sample variance of the gradient estimate to optimize the baseline function.

We perform experiments on a representative toy optimization problem and both auto-encoding variational inference and maximum likelihood estimation for discrete latent variable models, with one or multiple binary stochastic layers. Our extensive experiments show that the ARM estimator is unbiased, exhibits low variance, converges fast, has low computation, and provides state-of-the-art out-of-sample prediction performance for discrete latent variable models, suggesting the effectiveness of using the ARM estimator for gradient backpropagation through stochastic binary layers.

4.2 Main Result

In this section, we first present the key theorem of this chapter, and then provide its derivation. With this theorem, we summarize ARM gradient ascent for multivariate binary latent variables in Algorithm 3, as shown in

Appendix C, section C.1. Let us denote $\sigma(\phi) = e^{\phi}/(1+e^{\phi})$ as the sigmoid function and $\mathbf{1}_{[\cdot]}$ as an indicator function that equals to one if the argument is true and zero otherwise.

Theorem 4 (ARM)**.** *For a vector of binary random variables $\boldsymbol{z} = (z_1, \ldots, z_V)'$, the gradient of*

$$\mathcal{E}(\boldsymbol{\phi}) = \mathbb{E}_{\boldsymbol{z} \sim \prod_{v=1}^{V} \text{Bernoulli}(z_v; \sigma(\phi_v))}[f(\boldsymbol{z})] \tag{4.3}$$

with respect to $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_V)^T$, the logits of the Bernoulli probability parameters, can be expressed as

$$\nabla_{\boldsymbol{\phi}} \mathcal{E}(\boldsymbol{\phi}) = \mathbb{E}_{\boldsymbol{u} \sim \prod \text{Uniform}(0,1)} \left[\left(f(\mathbf{1}_{[\boldsymbol{u} > \sigma(-\boldsymbol{\phi})]}) - f(\mathbf{1}_{[\boldsymbol{u} < \sigma(\boldsymbol{\phi})]}) \right) (\boldsymbol{u} - 1/2) \right], \tag{4.4}$$

where $\mathbf{1}_{[\boldsymbol{u} > \sigma(-\boldsymbol{\phi})]} := \left(\mathbf{1}_{[u_1 > \sigma(-\phi_1)]}, \ldots, \mathbf{1}_{[u_V > \sigma(-\phi_V)]} \right)^T$.

4.2.1 Univariate ARM Estimator

Below we will first present the ARM estimator for a univariate binary latent variable, and then generalize it to a multivariate one. In the univariate case, we need to evaluate the gradient of $\mathcal{E}(\phi) = \mathbb{E}_{z \sim \text{Bernoulli}(\sigma(\phi))}[f(z)]$ with respect to ϕ, which has an analytic expression as

$$\nabla_{\phi} \mathcal{E}(\phi) = \nabla_{\phi}[\sigma(\phi) f(1) + \sigma(-\phi) f(0)] = \sigma(\phi)\sigma(-\phi)[f(1) - f(0)]. \tag{4.5}$$

Since $\int_0^{\sigma(\phi)} (1-2u)du = \sigma(\phi)\sigma(-\phi)$ and $\int_{\sigma(\phi)}^1 (1-2u)du = -\sigma(\phi)\sigma(-\phi)$, we can rewrite (4.5) as

$$\begin{aligned} \nabla_{\phi} \mathcal{E}(\phi) &= \textstyle\int_0^{\sigma(\phi)} f(1)(1-2u)du + \int_{\sigma(\phi)}^1 f(0)(1-2u)du \\ &= \mathbb{E}_{u \sim \text{Uniform}(0,1)}[f(\mathbf{1}_{[u < \sigma(\phi)]})(1-2u)]. \end{aligned} \tag{4.6}$$

We refer to (4.6) as the univariate augment-REINFORCE (AR) estimator.

Applying antithetic sampling [101] to the AR estimator in (4.6), with $\tilde{u} = 1 - u$, we have

$$\begin{aligned}\nabla_\phi \mathcal{E}(\phi) &= \mathbb{E}_{u(0,1)}[f(\mathbf{1}_{[u<\sigma(\phi)]})(1/2-u)] + \mathbb{E}_{\tilde{u}(0,1)}[f(\mathbf{1}_{[\tilde{u}<\sigma(\phi)]})(1/2-\tilde{u})] \\ &= \mathbb{E}_{u\sim\text{Uniform}(0,1)}[f(\mathbf{1}_{[u<\sigma(\phi)]})(1/2-u) + f(\mathbf{1}_{[\tilde{u}<\sigma(\phi)]})(1/2-\tilde{u})] \\ &= \mathbb{E}_{u\sim\text{Uniform}(0,1)}\left[\big(f(\mathbf{1}_{[u>\sigma(-\phi)]}) - f(\mathbf{1}_{[u<\sigma(\phi)]})\big)(u-1/2)\right], \qquad (4.7)\end{aligned}$$

which provides the proof for Theorem 4 for $V = 1$.

Note that since $\mathbb{E}_{u\sim\text{Uniform}(0,1)}[f(\mathbf{1}_{[u<\sigma(\phi)]})(1/2-u)] = -\mathbb{E}_{u\sim\text{Uniform}(0,1)}$ $[f(\mathbf{1}_{[u>\sigma(-\phi)]})(1/2-u)]$, we have

$$\mathbb{E}_{u\sim\text{Uniform}(0,1)}\left[\big(f(\mathbf{1}_{[u<\sigma(\phi)]}) + f(\mathbf{1}_{[u>\sigma(-\phi)]})\big)(1/2-u)\right] = 0,$$

subtracted which from the AR estimator in (4.6) leads to the ARM estimator in (4.7). For this reason, we can also consider the ARM estimator as the AR estimator subtracted by a zero-mean baseline function as

$$b(u) = \big(f(\mathbf{1}_{[u<\sigma(\phi)]}) + f(\mathbf{1}_{[u>\sigma(-\phi)]})\big)(1/2-u).$$

This baseline function is distinct from previously proposed ones in being parameterized by the function f itself over two correlated binary latent variables and satisfying $b(u) = -b(1-u)$. From this point of view, the ARM estimator can be considered as a "self-control" gradient estimator that exploits the function f itself to control the variance of Monte Carlo integration .

4.2.2 Multivariate Generalization

Although the ARM estimator for univariate binary is of little use in practice, as the true gradient, shown in (4.5), has an analytic expression, it serves as the foundation for generalizing it to multivariate settings. Let us denote $(\cdot)_{\backslash v}$ as a vector whose vth element is removed. For the expectation in (4.3), applying the univariate ARM estimator in (4.7) together with the law of total expectation, we have

$$\begin{aligned}\nabla_{\phi_v}\mathcal{E}(\boldsymbol{\phi}) &= \mathbb{E}_{\boldsymbol{z}_{\backslash v}\sim\prod_{j\neq v}\text{Bernoulli}(z_j;\sigma(\phi_j))}\big\{\nabla_{\phi_v}\mathbb{E}_{z_v\sim\text{Bernoulli}(\sigma(\phi_v))}[f(\boldsymbol{z})]\big\}\\ &= \mathbb{E}_{\boldsymbol{z}_{\backslash v}\sim\prod_{j\neq v}\text{Bernoulli}(z_j;\sigma(\phi_j))}\big\{\mathbb{E}_{u_v\sim\text{Uniform}(0,1)}\big[(u_v-1/2)\\ &\qquad\times\big(f(\boldsymbol{z}_{\backslash v},z_v=\mathbf{1}_{[u_v>\sigma(-\phi_v)]})-f(\boldsymbol{z}_{\backslash v},z_v=\mathbf{1}_{[u_v<\sigma(\phi_v)]})\big)\big]\big\}. \quad (4.8)\end{aligned}$$

Since $\boldsymbol{z}_{\backslash v}\sim\prod_{j\neq v}\text{Bernoulli}(z_j;\sigma(\phi_j))$ can be equivalently generated as $\boldsymbol{z}_{\backslash v}=\mathbf{1}_{[\boldsymbol{u}_{\backslash v}<\sigma(\boldsymbol{\phi}_{\backslash v})]}$ or as $\boldsymbol{z}_{\backslash v}=\mathbf{1}_{[\boldsymbol{u}_{\backslash v}>\sigma(-\boldsymbol{\phi}_{\backslash v})]}$, where $\boldsymbol{u}_{\backslash v}\sim\prod_{j\neq v}\text{Uniform}(u_j;0,1)$, exchanging the order of the two expectations in (4.8) and applying reparameterization, we conclude the proof for (4.4) of Theorem 4 with

$$\begin{aligned}\nabla_{\phi_v}\mathcal{E}(\boldsymbol{\phi}) &= \mathbb{E}_{u_v\sim\text{Uniform}(0,1)}\big\{(u_v-1/2)\,\mathbb{E}_{\boldsymbol{z}_{\backslash v}\sim\prod_{j\neq v}\text{Bernoulli}(z_j;\sigma(\phi_j))}\big[\\ &\qquad f(\boldsymbol{z}_{\backslash v},z_v=\mathbf{1}_{[u_v>\sigma(-\phi_v)]})-f(\boldsymbol{z}_{\backslash v},z_v=\mathbf{1}_{[u_v<\sigma(\phi_v)]})\big]\big\}\\ &= \mathbb{E}_{\boldsymbol{u}}\big[(u_v-1/2)f(\boldsymbol{z}_{\backslash v}=\mathbf{1}_{[\boldsymbol{u}_{\backslash v}>\sigma(-\boldsymbol{\phi}_{\backslash v})]},z_v=\mathbf{1}_{[u_v>\sigma(-\phi_v)]})\big]\\ &\quad-\mathbb{E}_{\boldsymbol{u}}\big[(u_v-1/2)f(\boldsymbol{z}_{\backslash v}=\mathbf{1}_{[\boldsymbol{u}_{\backslash v}<\sigma(\boldsymbol{\phi}_{\backslash v})]},z_v=\mathbf{1}_{[u_v<\sigma(\phi_v)]})\big]\\ &= \mathbb{E}_{\boldsymbol{u}\sim\prod\text{Uniform}(0,1)}\big[(u_v-1/2)\big(f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]})-f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]})\big)\big]. \quad (4.9)\end{aligned}$$

Alternatively, instead of generalizing the univariate ARM gradient as in (4.8) and (4.9), we can first do a multivariate generalization of the univariate

AR gradient in (4.6) as

$$\begin{aligned}\nabla_{\phi_v}\mathcal{E}(\boldsymbol{\phi}) &= \mathbb{E}_{\boldsymbol{z}_{\setminus v}\sim\prod_{j\neq v}\text{Bernoulli}(z_j;\sigma(\phi_j))}\big\{\nabla_{\phi_v}\mathbb{E}_{z_v\sim\text{Bernoulli}(\sigma(\phi_v)}[f(\boldsymbol{z})]\big\}\\ &= \mathbb{E}_{\boldsymbol{z}_{\setminus v}\sim\prod_{j\neq v}\text{Bernoulli}(z_j;\sigma(\phi_j))}\big\{\mathbb{E}_{u_v}\big[(1-2u_v)f(\boldsymbol{z}_{\setminus v}, z_v=\mathbf{1}_{[u_v<\sigma(\phi_v)]})\big]\big\}\\ &= \mathbb{E}_{\boldsymbol{u}\sim\prod_{v=1}^{V}\text{Uniform}(u_v;0,1)}\big[(1-2u_v)f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]})\big]. \qquad (4.10)\end{aligned}$$

The same as the derivation of the univariate ARM estimator, here we can arrive at (4.4) from (4.10) by either adding an antithetic sampling step, or subtracting the AR estimator by a baseline function as

$$\boldsymbol{b}(\boldsymbol{u}) = \big(f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]}) + f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]})\big)(1/2-\boldsymbol{u}), \qquad (4.11)$$

which has zero mean, satisfies $\boldsymbol{b}(\boldsymbol{u}) = -\boldsymbol{b}(1-\boldsymbol{u})$, and is distinct from previously proposed baselines in taking advantage of "self-control" for variance reduction and adding no extra parameters to learn.

4.2.3 Effectiveness of ARM for Variance Reduction

For the univariate case, we show below that the ARM estimator has smaller worst-case variance than REINFORCE does. The proof is deferred to Appendix C, section C.2.

Proposition 6 (Univariate gradient variance)**.** *For the objective function* $\mathbb{E}_{z\sim\text{Bernoulli}(\sigma(\phi))}[f(z)]$*, with a single Monte-Carlo sample* $u\sim\text{Uniform}(0,1)$ *or* $z\sim\text{Bernoulli}(\sigma(\phi))$*, the ARM gradient is expressed as* $g_{\text{ARM}}(u,\phi) = \big(f(\mathbf{1}_{[u>\sigma(-\phi)]}) - f(\mathbf{1}_{[u<\sigma(\phi)]})\big)(u-1/2)$*, and the REINFORCE gradient as* $g_{\text{R}}(z,\phi) = f(z)\nabla_\phi \log\text{Bernoulli}(z;\sigma(\phi)) = f(z)(z-\sigma(\phi))$*. Assuming* $f\geq 0$ *(or* $f\leq 0$*), then* $\frac{\sup_\phi \text{var}[g_{\text{ARM}}(u,\phi)]}{\sup_\phi \text{var}[g_{\text{R}}(u,\phi)]} \leq \frac{16}{25}(1-2\frac{f(0)}{f(0)+f(1)})^2 \leq \frac{16}{25}$.

In the general setting, with $\boldsymbol{u}^{(1)}, \ldots, \boldsymbol{u}^{(K)} \overset{iid}{\sim} \prod_{v=1}^{V} \text{Uniform}(0,1)$, we define the ARM estimate of $\nabla_{\phi_v} \mathcal{E}(\boldsymbol{\phi})$ with K Monte Carlo samples, denoted as $g_{\text{ARM}_K,v}$, and the AR estimate with $2K$ Monte Carlo samples, denoted as $g_{\text{AR}_{2K},v}$, using

$$g_{\text{ARM}_K,v} = \frac{1}{2K} \sum_{k=1}^{K} (g_v(\boldsymbol{u}^{(k)}) + g_v(1 - \boldsymbol{u}^{(k)})), \quad g_{\text{AR}_{2K},v} = \frac{1}{2K} \sum_{k=1}^{2K} g_v(\boldsymbol{u}^{(k)}), \tag{4.12}$$

where $g_v(\boldsymbol{u}^{(k)}) = f(\mathbf{1}_{[\boldsymbol{u}^{(k)} < \sigma(\boldsymbol{\phi})]})(1 - 2u_v^{(k)})$. Similar to the analysis in Owen [101], the amount of variance reduction brought by the ARM estimator can be reflected by the ratio as

$$\frac{\text{var}[g_{\text{ARM}_K,v}]}{\text{var}[g_{\text{AR}_{2K},v}]} = \frac{\text{var}[g_v(\boldsymbol{u})] - \text{Cov}(-g_v(\boldsymbol{u}), g_v(1 - \boldsymbol{u}))}{\text{var}[g_v(\boldsymbol{u})]} = 1 - \rho_v,$$
$$\rho_v = \text{Corr}(-g_v(\boldsymbol{u}), g_v(1 - \boldsymbol{u})).$$

Note $-g_v(\boldsymbol{u}) = f(\mathbf{1}_{[\boldsymbol{u} < \sigma(\boldsymbol{\phi})]})(2u_v - 1)$, $g_v(1 - \boldsymbol{u}) = f(\mathbf{1}_{[\boldsymbol{u} > \sigma(-\boldsymbol{\phi})]})(2u_v - 1)$, and $P(\mathbf{1}_{[u_v < \sigma(\phi_v)]} = \mathbf{1}_{[u_v > \sigma(-\phi_v)]}) = \sigma(|\phi_v|) - \sigma(-|\phi_v|)$. Therefore a strong positive correlation (*i.e.*, $\rho_v \to 1$) and hence noticeable variance reduction are likely, especially if ϕ_v moves far away from zero during training. Concretely, we have the following proposition.

Proposition 7 (Variance reduction)**.** *For the ARM estimate $g_{\text{ARM}_K,v}$ and AR estimate $g_{\text{AR}_{2K},v}$ shown in* (4.12)*, the variance of $g_{\text{ARM}_K,v}$ is guaranteed to be lower than that of $g_{\text{AR}_K,v}$; moreover, if $f \geq 0$ (or $f \leq 0$), then the variance of $g_{\text{ARM}_K,v}$ is guaranteed to be lower than that of $g_{\text{AR}_{2K},v}$.*

We show below that under the anti-symmetric constraint $\boldsymbol{b}(\boldsymbol{u}) = -\boldsymbol{b}(1 - \boldsymbol{u})$, which implies that $\mathbb{E}_{\boldsymbol{u} \sim \prod_{v=1}^{V} \text{Uniform}(u_v;0,1)}[\boldsymbol{b}(\boldsymbol{u})]$ is a vector of

zeros, Equation (4.11) is the optimal baseline function to be subtracted from the AR estimator for variance reduction. The proof is deferred to Appendix C, section C.2.

Proposition 8 (Optimal anti-symmetric baseline)**.** *For the gradient of ϕ with respect to $\mathbb{E}_{\boldsymbol{z}\sim q_{\boldsymbol{\phi}}(\boldsymbol{z})}[f(\boldsymbol{z})]$, the optimal anti-symmetric baseline function to be subtracted from the AR estimator $g_{\mathrm{AR}}(\boldsymbol{u}) = f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]})(1-2\boldsymbol{u})$, which minimizes the variance of Monte Carlo integration, can be expressed as*

$$\arg\min_{b_v(\boldsymbol{u})\in\mathcal{B}} \mathrm{var}[g_{\mathrm{AR},v}(\boldsymbol{u}) - b_v(\boldsymbol{u})] = \frac{1}{2}(g_{\mathrm{AR},v}(\boldsymbol{u}) - g_{\mathrm{AR},v}(1-\boldsymbol{u})), \tag{4.13}$$

where $\mathcal{B} = \{\boldsymbol{b}(\boldsymbol{u}) : b_v(\boldsymbol{u}) = -b_v(1-\boldsymbol{u})$ for all $v\}$ is the set of all zero-mean anti-symmetric baseline functions. Note the optimal baseline function shown in (4.13) *is exactly the same as* (4.11)*, which is subtracted from the AR estimator in* (4.10) *to arrive at the ARM estimator in* (4.4)*.*

Corollary 2 (Lower variance than constant baseline)**.** *The optimal anti-symmetric baseline function for the AR estimator, as shown in* (4.13) *and also in* (4.11)*, leads to lower estimation variance than any constant based baseline function as $b_v(\boldsymbol{u}) = c_v(1/2 - u_v)$, where c_v is a dimension-specific constant whose value can be optimized for variance reduction.*

4.3 Applications in Discrete Optimization

A latent variable model with multiple stochastic hidden layers can be constructed as

$$\boldsymbol{x} \sim p_{\boldsymbol{\theta}_0}(\boldsymbol{x}\,|\,\boldsymbol{b}_1),\ \boldsymbol{b}_1 \sim p_{\boldsymbol{\theta}_1}(\boldsymbol{b}_1\,|\,\boldsymbol{b}_2), \ldots, \boldsymbol{b}_t \sim p_{\boldsymbol{\theta}_t}(\boldsymbol{b}_t\,|\,\boldsymbol{b}_{t+1}), \ldots, \boldsymbol{b}_T \sim p_{\boldsymbol{\theta}_T}(\boldsymbol{b}_T),$$

whose joint likelihood given the distribution parameters $\boldsymbol{\theta}_{0:T} = \{\boldsymbol{\theta}_0, \ldots, \boldsymbol{\theta}_T\}$ is expressed as

$$p(\boldsymbol{x}, \boldsymbol{b}_{1:T} \,|\, \boldsymbol{\theta}_{0:T}) = p_{\boldsymbol{\theta}_0}(\boldsymbol{x} \,|\, \boldsymbol{b}_1) \Big[\prod\nolimits_{t=1}^{T-1} p_{\boldsymbol{\theta}_t}(\boldsymbol{b}_t \,|\, \boldsymbol{b}_{t+1}) \Big] p_{\boldsymbol{\theta}_T}(\boldsymbol{b}_T). \tag{4.14}$$

In comparison to deterministic feedforward neural networks, stochastic ones can represent complex distributions and show natural resistance to overfitting [98, 128, 136, 110, 47, 136]. However, the training of the network, especially if there are stochastic discrete layers, is often much more challenging. Below we show for both auto-encoding variational inference and maximum likelihood estimation, how to apply the ARM estimator for gradient backpropagation in stochastic binary networks.

4.3.1 ARM for Variational Auto-Encoder

For auto-encoding variational inference [69, 117], we construct a variational distribution as

$$q_{\boldsymbol{w}_{1:T}}(\boldsymbol{b}_{1:T} \,|\, \boldsymbol{x}) = q_{\boldsymbol{w}_1}(\boldsymbol{b}_1 \,|\, \boldsymbol{x}) \Big[\prod\nolimits_{t=1}^{T-1} q_{\boldsymbol{w}_{t+1}}(\boldsymbol{b}_{t+1} \,|\, \boldsymbol{b}_t) \Big], \tag{4.15}$$

with which the ELBO can be expressed as

$$\mathcal{E}(\boldsymbol{w}_{1:T}) = \mathbb{E}_{\boldsymbol{b}_{1:T} \sim q_{\boldsymbol{w}_{1:T}}(\boldsymbol{b}_{1:T} \,|\, \boldsymbol{x})} \left[f(\boldsymbol{b}_{1:T}) \right], \text{ where} \tag{4.16}$$

$$f(\boldsymbol{b}_{1:T}) = \log p_{\boldsymbol{\theta}_0}(\boldsymbol{x} \,|\, \boldsymbol{b}_1) + \log p_{\boldsymbol{\theta}_{1:T}}(\boldsymbol{b}_{1:T}) - \log q_{\boldsymbol{w}_{1:T}}(\boldsymbol{b}_{1:T} \,|\, \boldsymbol{x}). \tag{4.17}$$

Proposition 9 (ARM backpropagation)**.** *For a stochastic binary network with T binary stochastic hidden layers, constructing a variational auto-encoder*

(VAE) defined with $\boldsymbol{b}_0 = \boldsymbol{x}$ *and*

$$q_{\boldsymbol{w}_t}(\boldsymbol{b}_t \,|\, \boldsymbol{b}_{t-1}) = \text{Bernoulli}(\boldsymbol{b}_t; \sigma(\mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1}))) \tag{4.18}$$

for $t = 1, \ldots, T$*, the gradient of the ELBO with respect to* $\boldsymbol{w}_t$ *is*

$$\nabla_{\boldsymbol{w}_t} \mathcal{E}(\boldsymbol{w}_{1:T}) = \mathbb{E}_{q(\boldsymbol{b}_{1:t-1})}\left[\mathbb{E}_{\boldsymbol{u}_t}[f_\Delta(\boldsymbol{u}_t, \mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1}), \boldsymbol{b}_{1:t-1})(\boldsymbol{u}_t - \frac{1}{2})]\nabla_{\boldsymbol{w}_t}\mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1})\right],$$

where $f_\Delta = \mathbb{E}_{\boldsymbol{b}_{t+1:T}\sim q(\boldsymbol{b}_{t+1:T}\,|\,\boldsymbol{b}_t),\ \boldsymbol{b}_t=\mathbf{1}_{[\boldsymbol{u}_t>\sigma(-\mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1}))]}}[f(\boldsymbol{b}_{1:T})]$

$$- \mathbb{E}_{\boldsymbol{b}_{t+1:T}\sim q(\boldsymbol{b}_{t+1:T}\,|\,\boldsymbol{b}_t),\ \boldsymbol{b}_t=\mathbf{1}_{[\boldsymbol{u}_t<\sigma(\mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1}))]}}[f(\boldsymbol{b}_{1:T})]. \tag{4.19}$$

The gradient presented in (4.19) can be estimated with a single Monte Carlo sample as

$$\hat{f}_\Delta(\boldsymbol{u}_t, \mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1}), \boldsymbol{b}_{1:t-1}) = \begin{cases} 0, & \text{if } \boldsymbol{b}_t^{(1)} = \boldsymbol{b}_t^{(2)}, \\ f(\boldsymbol{b}_{1:t-1}, \boldsymbol{b}_{t:T}^{(1)}) - f(\boldsymbol{b}_{1:t-1}, \boldsymbol{b}_{t:T}^{(2)}), & \text{otherwise,} \end{cases}$$

where $\boldsymbol{b}_t^{(1)} = \mathbf{1}_{[\boldsymbol{u}_t>\sigma(-\mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1}))]}$, $\boldsymbol{b}_{t+1:T}^{(1)} \sim q(\boldsymbol{b}_{t+1:T}\,|\,\boldsymbol{b}_t^{(1)})$, $\boldsymbol{b}_t^{(2)} = \mathbf{1}_{[\boldsymbol{u}_t<\sigma(\mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1}))]}$, and $\boldsymbol{b}_{t+1:T}^{(2)} \sim q(\boldsymbol{b}_{t+1:T}\,|\,\boldsymbol{b}_t^{(2)})$. The proof of Proposition 9 is provided in Appendix C, section C.2. Suppose the computation complexity of vanilla REINFORCE for a stochastic hidden layer is $\mathcal{O}(1)$, which involves a single evaluation of the function f and gradient backpropagation as $\nabla_{\boldsymbol{w}_t}\mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1})$, then for a T-stochastic-hidden-layer network, the computation complexity of vanilla REINFORCE is $\mathcal{O}(T)$. By contrast, if evaluating f is much less expensive in computation than gradient backpropagation, then the ARM estimator also has $\mathcal{O}(T)$ complexity, whereas if evaluating f dominates gradient backpropagation in computation, then its worst-case complexity is $\mathcal{O}(2T)$.

4.3.2 ARM for Maximum Likelihood Estimation

For maximum likelihood estimation, the log marginal likelihood can be expressed as

$$\begin{aligned}\log p_{\boldsymbol{\theta}_{0:T}}(\boldsymbol{x}) &= \log \mathbb{E}_{\boldsymbol{b}_{1:T}\sim p_{\boldsymbol{\theta}_{1:T}}(\boldsymbol{b}_{1:T})}[p_{\boldsymbol{\theta}_0}(\boldsymbol{x}\,|\,\boldsymbol{b}_1)] \\ &\geq \mathcal{E}(\boldsymbol{\theta}_{1:T}) = \mathbb{E}_{\boldsymbol{b}_{1:T}\sim p_{\boldsymbol{\theta}_{1:T}}(\boldsymbol{b}_{1:T})}[\log p_{\boldsymbol{\theta}_0}(\boldsymbol{x}\,|\,\boldsymbol{b}_1)]. \end{aligned} \tag{4.20}$$

Generalizing Proposition 9 leads to the following proposition.

Proposition 10. *For a stochastic binary network defined as*

$$p_{\boldsymbol{\theta}_t}(\boldsymbol{b}_t\,|\,\boldsymbol{b}_{t+1}) = \text{Bernoulli}(\boldsymbol{b}_t; \sigma(\mathcal{T}_{\boldsymbol{\theta}_t}(\boldsymbol{b}_{t+1}))), \tag{4.21}$$

the gradient of the lower bound in (4.20) *with respect to* $\boldsymbol{\theta}_t$ *can be expressed as*

$$\nabla_{\boldsymbol{\theta}_t}\mathcal{E}(\boldsymbol{\theta}_{1:T}) = \mathbb{E}_{p(\boldsymbol{b}_{t+1:T})}\left[\mathbb{E}_{\boldsymbol{u}_t}[f_\Delta(\boldsymbol{u}_t, \mathcal{T}_{\boldsymbol{\theta}_t}(\boldsymbol{b}_{t+1}), \boldsymbol{b}_{t+1:T})(\boldsymbol{u}_t - 1/2)]\nabla_{\boldsymbol{\theta}_t}\mathcal{T}_{\boldsymbol{\theta}_t}(\boldsymbol{b}_{t+1})\right],$$

where $f_\Delta = \mathbb{E}_{\boldsymbol{b}_{1:t-1}\sim p(\boldsymbol{b}_{1:t-1}\,|\,\boldsymbol{b}_t),\ \boldsymbol{b}_t=\mathbf{1}_{[\boldsymbol{u}_t>\sigma(-\mathcal{T}_{\boldsymbol{\theta}_t}(\boldsymbol{b}_{t+1}))]}}[\log p_{\boldsymbol{\theta}_0}(\boldsymbol{x}\,|\,\boldsymbol{b}_1)]$

$$- \mathbb{E}_{\boldsymbol{b}_{1:t-1}\sim p(\boldsymbol{b}_{1:t-1}\,|\,\boldsymbol{b}_t),\ \boldsymbol{b}_t=\mathbf{1}_{[\boldsymbol{u}_t<\sigma(\mathcal{T}_{\boldsymbol{\theta}_t}(\boldsymbol{b}_{t+1}))]}}[\log p_{\boldsymbol{\theta}_0}(\boldsymbol{x}\,|\,\boldsymbol{b}_1)]. \tag{4.22}$$

4.4 Experimental Results

To illustrate the working mechanism of the ARM estimator, related to Tucker et al. [147] and Grathwohl et al. [45], we consider learning ϕ to maximize

$$\mathcal{E}(\phi) = \mathbb{E}_{z\sim\text{Bernoulli}(\sigma(\phi))}[(z - p_0)^2],\ \text{where } p_0 \in \{0.49, 0.499, 0.501, 0.51\}.$$

The optimal solution is $\sigma(\phi) = \mathbf{1}_{[p_0<0.5]}$. The closer p_0 is to 0.5, the more challenging the optimization becomes. We compare both the AR and ARM

estimators to the true gradient as

$$g_\phi = (1 - 2p_0)\sigma(\phi)(1 - \sigma(\phi)) \tag{4.23}$$

and three previously proposed unbiased estimators, including REINFORCE, REBAR [147], and RELAX [45]. Since RELAX is closely related to REBAR in introducing stochastically estimated control variates to improve REINFORCE, and clearly outperforms REBAR in our experiments for this toy problem (as also shown in Grathwohl et al. [45] for $p_0 = 0.49$), we omit the results of REBAR for brevity. With a single random sample $u \sim \text{Uniform}(0, 1)$ for Monte Carlo integration, the REINFORCE and AR gradients can be expressed as

$$g_{\phi,\text{REINFORCE}} = (\mathbf{1}_{[u<\sigma(\phi)]} - p_0)^2(\mathbf{1}_{[u<\sigma(\phi)]} - \sigma(\phi)),\ g_{\phi,\text{AR}} = (\mathbf{1}_{[u<\sigma(\phi)]} - p_0)^2(1 - 2u),$$

while the ARM gradient can be expressed as

$$g_{\phi,\text{ARM}} = \left[(\mathbf{1}_{[u>\sigma(-\phi)]} - p_0)^2 - (\mathbf{1}_{[u<\sigma(\phi)]} - p_0)^2\right](u - 1/2).$$

See Grathwohl et al. [45] for the details on RELAX.

In Figure 4.1, p_0 takes values 0.49, 0.499, 0.501, 0.51; the optimal solution is $\sigma(\phi) = \mathbf{1}(p_0 < 0.5)$. The top two rows are the trace plots of the true/estimated gradients $\nabla_\phi \mathcal{E}(\phi)$ and estimated Bernoulli probability parameters $\sigma(\phi)$, with ϕ updated via gradient ascent. The bottom row is the gradient variances for $p_0 = 0.49$, estimated using $K = 5000$ Monte Carlo samples at each iteration. It demonstrates that the REINFORCE gradients have large variances. Consequently, a REINFORCE based gradient ascent algorithm may

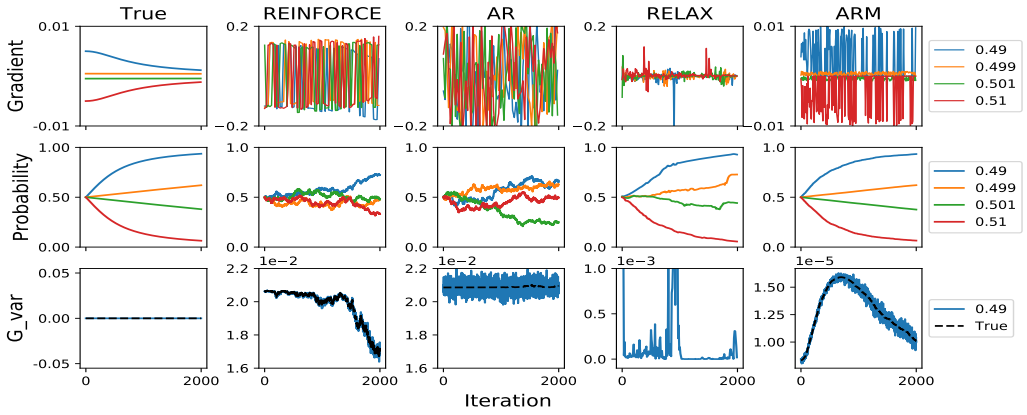


Figure 4.1: Comparison of a variety of gradient estimators in maximizing $\mathcal{E}(\phi) = \mathbb{E}_{z\sim\text{Bernoulli}(\sigma(\phi))}[(z - p_0)^2]$ via gradient ascent.

diverge if the gradient ascent stepsize is not sufficiently small. For example, when $p_0 = 0.501$, the optimal value for the Bernoulli probability $\sigma(\phi)$ is 0, but the algorithm with 0.1 as the stepsize infers it to be close to 1 at the end of 2000 iterations of a random trial. The AR estimator behaves similarly as REINFORCE does. By contrast, both RELAX and ARM exhibit clearly lower estimation variance. It is interesting to note that the trace plots of the estimated probability $\sigma(\phi)$ with the univariate ARM estimator almost exactly match these with the true gradients, despite that the trace plots of the ARM gradients are distinct from these of the true gradients. More specifically, while the true gradients smoothly evolve over iterations, the univariate ARM gradients are characterized by zeros and random spikes; this distinct behavior is expected by examining (C.1) in Appendix C, section C.2, which suggests that at any given iteration, the univariate ARM gradient based on a single Monte Carlo sample is either exactly zero, which happens with probability

$\sigma(|\phi|) - \sigma(-|\phi|)$, or taking $|[f(1) - f(0)](1/2 - u)|$ as its absolute value. These observations suggest that by adjusting the frequencies and amplitudes of spike gradients, the univariate ARM estimator very well approximates the behavior of the true gradient for learning with gradient ascent.

4.4.1 Discrete Variational Auto-Encoders

To optimize a variational auto-encoder (VAE) for a discrete latent variable model, existing solutions often rely on biased but low-variance stochastic gradient estimators [10, 64], unbiased but high-variance ones [90], or unbiased REINFORCE combined with computationally expensive baselines, whose parameters are estimated by minimizing the sample variance of the estimator with SGD [147, 45]. By contrast, the ARM estimator exhibits low variance and is unbiased, efficient to compute, and simple to implement.

For discrete VAEs, we compare ARM with a variety of representative stochastic gradient estimators for discrete latent variables, including Wake-Sleep [54], NVIL [90], LeGrad [143], MuProp [47], Concrete (Gumbel-Softmax) [64, 84], REBAR [45], and RELAX [147]. Following the settings in Tucker et al. [147] and Grathwohl et al. [45], for the encoder defined in (4.14) and decoder defined in (4.15), we consider three different network architectures, as summarized in Table C.1, including "Nonlinear" that has one stochastic but two Leaky-ReLU [83] deterministic hidden layers, "Linear" that has one stochastic hidden layer, and "Linear two layers" that has two stochastic hidden layers. We consider a widely used binarization [121, 77, 161], referred to as MNIST-static,

Table 4.1: Test negative log-likelihoods of discrete VAEs trained with a variety of stochastic gradient estimators on MNIST-static and OMNIGLOT.

(a) MNIST-static

Linear		Nonlinear		Two layers	
Algorithm	$-\log p(x)$	Algorithm	$-\log p(x)$	Algorithm	$-\log p(x)$
AR	$= 164.1$	AR	$= 114.6$	AR	$= 162.2$
REINFORCE	$= 170.1$	REINFORCE	$= 114.1$	REINFORCE	$= 159.2$
Wake-Sleep*	$= 120.8$	Wake-Sleep*	-	Wake-Sleep*	$= 107.7$
NVIL *	$= 113.1$	NVIL *	$= 102.2$	NVIL*	$= 99.8$
LeGrad	≤ 117.5	LeGrad	-	LeGrad	-
MuProp†	≤ 113.0	MuProp⋆	$= 99.1$	MuProp†	≤ 100.4
Concrete⋆	$= 107.2$	Concrete⋆	$= 99.6$	Concrete⋆	$= 95.6$
REBAR⋆	$= 107.7$	REBAR⋆	$= 100.7$	REBAR⋆	$=$ **95.7**
RELAX‡	≤ 113.6	RELAX‡	≤ 119.2	RELAX‡	≤ 100.9
ARM	$=$ **107.2** $\pm$ **0.1**	ARM	$=$ **98.4** $\pm$ **0.3**	ARM	$= 96.7 \pm 0.3$

(b) OMNIGLOT

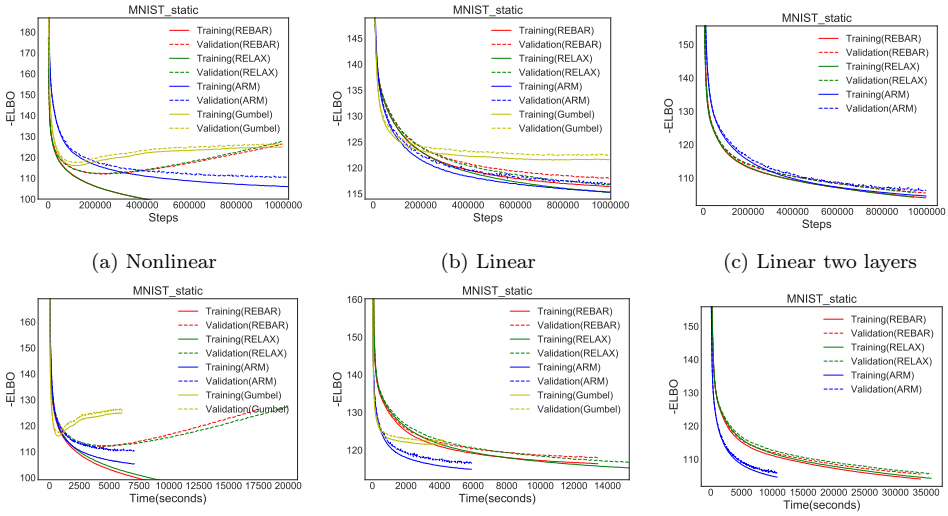
Linear		Nonlinear		Two layers	
Algorithm	$-\log p(x)$	Algorithm	$-\log p(x)$	Algorithm	$-\log p(x)$
NVIL*	$= 117.6$	NVIL*	$=$ **116.6**	NVIL*	$= 111.4$
MuProp⋆	$= 117.6$	MuProp⋆	$= 117.5$	MuProp⋆	$= 111.2$
Concrete⋆	$= 117.7$	Concrete⋆	$= 116.7$	Concrete⋆	$= 111.3$
REBAR⋆	$= 117.7$	REBAR⋆	$= 118.0$	REBAR⋆	$= 110.8$
RELAX‡	≤ 122.1	RELAX‡	≤ 128.2	RELAX‡	≤ 115.4
ARM	$=$ **115.8** $\pm$ **0.2**	ARM	$= 117.6 \pm 0.4$	ARM	$=$ **109.8** $\pm$ **0.3**

making our numerical results directly comparable to those reported in the literature. In addition to MNIST-static, we also consider MNIST-threshold [149], which binarizes MNIST by thresholding each pixel value at 0.5, and the binarized OMNIGLOT dataset.

We train discrete VAEs with 200 conditionally *iid* Bernoulli random variables as the hidden units of each stochastic binary layer. We maximize a single-Monte-Carlo-sample ELBO using Adam [68], with the learning rate selected from $\{5, 1, 0.5\} \times 10^{-4}$ by the validation set. We set the batch size as 50 for MNIST and 25 for OMNIGLOT. For each dataset, using its default training/validation/testing partition, we train all methods on the training set, calculate the validation log-likelihood for every epoch, and report the test negative log-likelihood when the validation negative log-likelihood reaches its minimum within a predefined maximum number of iterations.

We summarize the test negative log-likelihoods in Table 4.1 for MNIST-static. The symbols $*$, $\star$, †, ‡ represent the results reported in Mnih and Gregor [90], Tucker et al. [147], Gu et al. [47], and Grathwohl et al. [45], respectively. The results for LeGrad [143] are obtained by running the code provided by the authors. We report the results of ARM using the sample mean and standard deviation over five independent trials with random initializations.

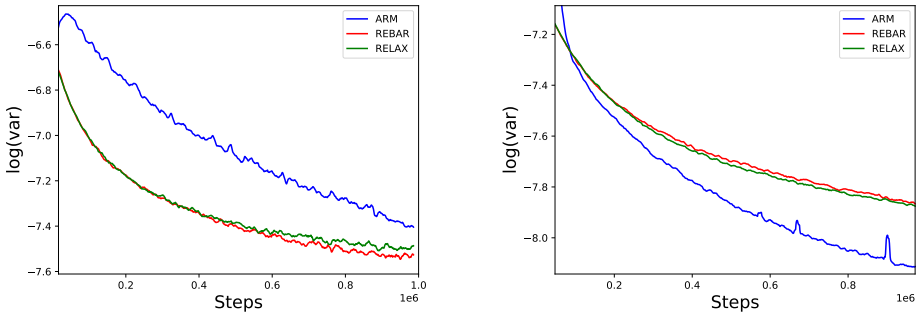
We also provide trace plots of the training and validation negative ELBOs on MNIST-static with respect to training iterations and wall clock time (on Tesla-K40 GPU) in Figure 4.2, and these on MNIST-threshold and OMNIGLOT in Figures 4.4 and 4.5, respectively.



(a) Nonlinear (b) Linear (c) Linear two layers

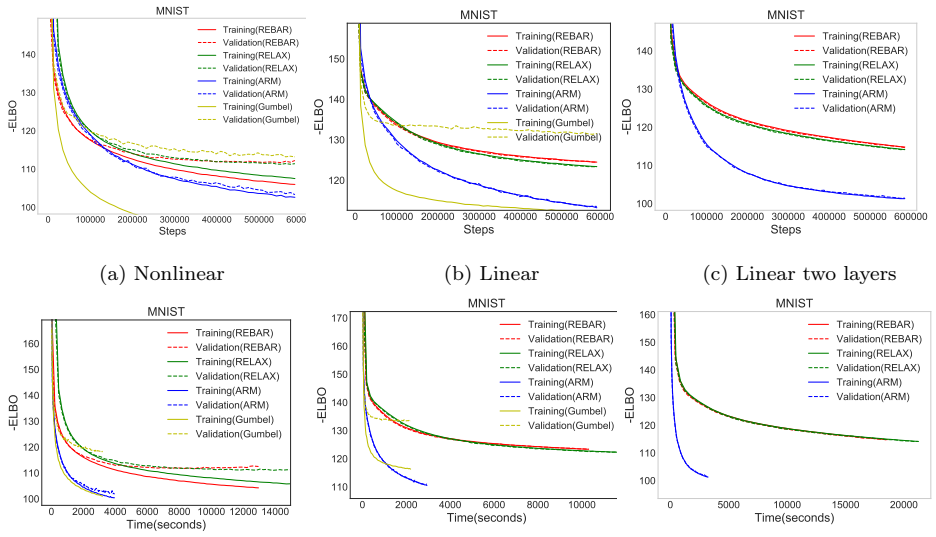
(d) Nonlinear (e) Linear (f) Linear two layers

Figure 4.2: Training and validation negative ELBOs on MNIST-static with respect to the training iterations and the wall clock time.



Nonlinear Linear

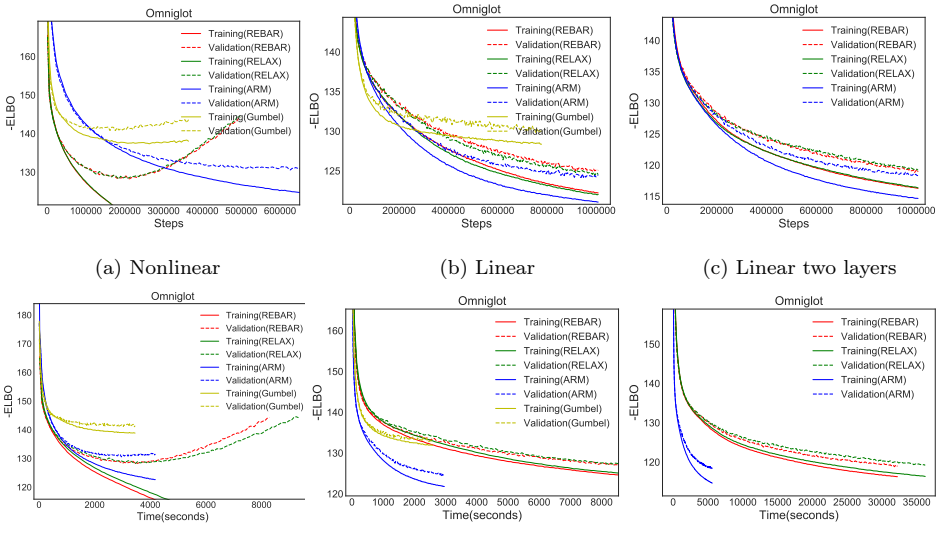
Figure 4.3: Trace plots of the log variance of the gradient estimators on the MNIST-static data for "Nonlinear" and "Linear" network architectures.



(a) Nonlinear (b) Linear (c) Linear two layers

(d) Nonlinear (e) Linear (f) Linear two layers

Figure 4.4: Training and validation negative ELBOs on MNIST-threshold with respect to the training iterationsand the wall clock time.



(a) Nonlinear (b) Linear (c) Linear two layers

(d) Nonlinear (e) Linear (f) Linear two layers

Figure 4.5: Training and validation negative ELBOs on OMNIGLOT with respect to the training iterations, shown in the top row, and with respect to the wall clock times on Tesla-K40 GPU, shown in the bottom row.

For these trace plots, for a fair comparison of convergence speed between different algorithms, we use publicly available code and setting the learning rate of ARM the same as that selected by RELAX in Grathwohl et al. [45]. Note as shown in Figures 4.2(a,d) and 4.5(a,d), both REBAR and RELAX exhibit clear signs of overfitting on both MNIST-static and Omniglot using the "Nonlinear" architecture; as ARM runs much faster per iteration than both of them and do not exhibit overfitting given the same number of iterations, we allow ARM to run more stochastic gradient ascent steps under these two scenarios to check whether it will eventually overfit the training set.

These results show that ARM provides superior performance in delivering not only fast convergence, but also low negative log-likelihoods and negative ELBOs on both the validation and test sets, with low computational cost, for all three different network architectures. In comparison to the vanilla REINFORCE on MNIST-static, as shown in Table 4.1 (a), ARM achieves significantly lower test negative log-likelihoods, which can be explained by having much lower variance in its gradient estimation, while only costing 20% to 30% more computation time to finish the same number of iterations.

The trace plots in Figures 4.2, 4.4, and 4.5 show that ARM achieves its objective better or on a par with the competing methods in all three different network architectures. In particular, the performance of ARM on MNIST-threshold is significantly better, suggesting ARM is more robust, better resists overfitting, and has better generalization ability. On both MNIST-static and OMNIGLOT, with the "Nonlinear" network architecture, both REBAR and

RELAX exhibit severe overfitting, which could be caused by their training procedure, which updates the parameters of the baseline function by minimizing the sample variance of the gradient estimator using SGD. For less overfitting linear and two-stochastic-layer networks, ARM overall performs better than both REBAR and RELAX and runs significantly faster (about 6-8 times faster) in terms of the computation time per iteration.

To understand why ARM has the best overall performance, we examine the trace plots of the logarithm of the estimated variance of gradient estimates in Figure 4.3. The variance of the gradient of each element is estimated by performing exponential smoothing, with the smoothing factor as 0.999, on its first two moments. On the MNIST-static dataset with the "Nonlinear" network, the left subplot of Figure 4.3 shows that both REBAR and RELAX exhibit lower variance than ARM does for their single-Monte-Carlo-sample based gradient estimates; however, the corresponding trace plots of the validation negative ELBOs, shown in Figure 4.2(a), suggest they both severely overfit the training data as the learning progresses; our hypothesis for this phenomenon is that REBAR and RELAX may favor suboptimal solutions that are associated with lower gradient variance; in other words, they may have difficulty in converging to local optimal solutions that are associated with high gradient variance. For the "Linear" network architecture, the right subplot of Figure 4.3 shows that ARM exhibits lower variance for its gradient estimate than both REBAR and RELAX do, and Figure 4.2(b) shows that none of them exhibit clear signs of overfitting; this observation could be used to explain why ARM results in

the best convergence for both the training and validation negative ELBOs, as shown in Figure 4.2(b).

4.4.2 Maximizing Likelihood for a Stochastic Binary Network

Denoting $\boldsymbol{x}_l, \boldsymbol{x}_u \in \mathbb{R}^{394}$ as the lower and upper halves of an MNIST digit, respectively, we consider a standard benchmark task of estimating the conditional distribution $p_{\boldsymbol{\theta}_{0:2}}(\boldsymbol{x}_l \,|\, \boldsymbol{x}_u)$ [110, 10, 47, 64, 147], using a stochastic binary network with two stochastic binary hidden layers, expressed as

$$\boldsymbol{x}_l \sim \text{Bernoulli}(\sigma(\mathcal{T}_{\boldsymbol{\theta}_0}(\boldsymbol{b}_1))),\ \boldsymbol{b}_1 \sim \text{Bernoulli}(\sigma(\mathcal{T}_{\boldsymbol{\theta}_1}(\boldsymbol{b}_2))),\ \boldsymbol{b}_2 \sim \text{Bernoulli}(\sigma(\mathcal{T}_{\boldsymbol{\theta}_2}(\boldsymbol{x}_u))).$$

We set the network structure as 392-200-200-392, which means both $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ are 200 dimensional binary vectors and the transformation $\mathcal{T}_{\boldsymbol{\theta}}$ are linear so the results are directly comparable with those in Jang et al. [64]. We approximate $\log p_{\boldsymbol{\theta}_{0:2}}(\boldsymbol{x}_l \,|\, \boldsymbol{x}_u)$ with $\log \frac{1}{K} \sum_{k=1}^{K} \text{Bernoulli}(\boldsymbol{x}_l; \sigma(\mathcal{T}_{\boldsymbol{\theta}_0}(\boldsymbol{b}_1^{(k)})))$, where $\boldsymbol{b}_1^{(k)} \sim \text{Bernoulli}(\sigma(\mathcal{T}_{\boldsymbol{\theta}_1}(\boldsymbol{b}_2^{(k)})))$, $\boldsymbol{b}_2^{(k)} \sim \text{Bernoulli}(\sigma(\mathcal{T}_{\boldsymbol{\theta}_2}(\boldsymbol{x}_u)))$. We perform training with $K = 1$, which can also be considered as optimizing on a single-Monte-Carlo-sample estimate of the lower bound of the log marginal likelihood shown in (4.20). We use Adam [68], with the learning rate set as 10^{-4}, mini-batch size as 100, and number of epochs for training as 2000. Given the inferred point estimate of $\boldsymbol{\theta}_{0:2}$ after training, we evaluate the accuracy of conditional density estimation by estimating the negative log-likelihood as $-\log p_{\boldsymbol{\theta}_{0:2}}(\boldsymbol{x}_l \,|\, \boldsymbol{x}_u)$, averaging over the test set using $K = 1000$. We show example results of predicting the activation probabilities of the pixels of $\boldsymbol{x}_l$ given $\boldsymbol{x}_u$ in Figure C.1 of the Appendix C.

Table 4.2: Comparison of the test negative log-likelihoods between ARM and various gradient estimators, for the MNIST conditional distribution estimation benchmark task.

Gradient estimator	ARM	ST	DARN	Annealed ST	ST Gumbel-S.	SF	MuProp
$-\log p(\boldsymbol{x}_l \mid \boldsymbol{x}_u)$	**57.9 ± 0.1**	58.9	59.7	58.7	59.3	72.0	58.9

As shown in Table 4.2, optimizing a stochastic binary network with the ARM estimator, which is unbiased and computationally efficient, achieves the lowest test negative log-likelihood, outperforming previously proposed biased stochastic gradient estimators, as reported in [64], on similarly structured stochastic networks, including DARN [46], straight through (ST) [10], slope-annealed ST [21], and ST Gumbel-softmax [64], and unbiased ones, including score-function (SF) and MuProp [47].

4.5 Concluding Remarks

To train a discrete latent variable model with one or multiple stochastic binary layers, we propose the augment-REINFORCE-merge (ARM) estimator to provide unbiased and low-variance gradient estimates of the parameters of Bernoulli distributions. With a single Monte Carlo sample, the estimated gradient is the product of uniform random noises and the difference of a function of two vectors of correlated binary latent variables. Without relying on estimating a baseline function with extra learnable parameters for variance reduction, it maintains efficient computation and avoids increasing the risk of overfitting. Applying the ARM gradient leads to not only fast convergence,

but also low test negative log-likelihoods (and low test negative evidence lower bounds for variational inference), on both auto-encoding variational inference and maximum likelihood estimation for stochastic binary feedforward neural networks. Some natural extensions of the proposed ARM estimator include generalizing it to multivariate categorical latent variables [164], combining it with a baseline or local-expectation based variance reduction methods, applying it to reinforcement learning with discrete action space [137, 167], and applying it to natural language processing [30].

Chapter 5

Meta-Learning with Variational Regularization

This chapter, based on the publication [165], studies the generalization problem of meta-learning. Meta-learning is a popular technique for leveraging data from previous tasks to enable efficient learning of new tasks. However, we find that most meta-learning algorithms implicitly require that the meta-training tasks be *mutually-exclusive*, such that no single model can solve all of the tasks at once. This requirement means that the user must take great care in designing the tasks, for example by shuffling labels or removing task identifying information from the inputs. In some domains, this makes meta-learning entirely inapplicable. In this chapter, we address this challenge by designing a meta-regularization objective using variational methods that places precedence on data-driven adaptation. This causes the meta-learner to decide what must

The content in this chapter was published in [165], Yin, Mingzhang, George Tucker, Mingyuan Zhou, Sergey Levine and Chelsea Finn. "Meta-Learning without Memorization". In International Conference on Learning Representations, 2020. I observed the problem, formalized the problem definition and designed the algorithms with the co-authors. I provided theoretical analysis in Section 5.4.1, implemented the methodology and wrote the draft paper. Prof. Finn, Prof. Levine and Dr. Tucker helped in defining the problem, adjusting the algorithm, creating the dataset, and the revision of the manuscript. Dr. Tucker provided theoretical analysis in Section 5.4.2. Prof. Zhou helped in the draft revision.

be learned from the task training data and what should be inferred from the task testing input. We demonstrate its applicability to both contextual and gradient-based meta-learning algorithms, and apply it in practical settings where applying standard meta-learning has been difficult.

5.1 Meta-Learning and Task Overfitting

The ability to learn new concepts and skills with small amounts of data is a critical aspect of intelligence that many machine learning systems lack. Meta-learning [129] has emerged as a promising approach for enabling systems to quickly learn new tasks by building upon experience from previous related tasks [139, 71, 124, 114, 32]. Meta-learning accomplishes this by explicitly optimizing for few-shot generalization across a set of meta-training tasks. The meta-learner is trained such that, after being presented with a small task training set, it can accurately make predictions on test datapoints for that meta-training task.

While these methods have shown promising results, current methods require careful design of the meta-training tasks to prevent a subtle form of *task overfitting*, distinct from standard overfitting in supervised learning. If the task can be accurately inferred from the test input alone, then the task training data can be ignored while still achieving low meta-training loss. In effect, the model will collapse to one that makes zero-shot decisions. This presents an opportunity for overfitting where the meta-learner generalizes on meta-training tasks, but fails to adapt when presented with training data from

novel tasks. We call this form of overfitting the *memorization problem* in meta-learning because the meta-learner memorizes a function that solves all of the meta-training tasks, rather than learning to adapt.

Existing meta-learning algorithms implicitly resolve this problem by carefully designing the meta-training tasks such that no single model can solve all tasks zero-shot; we call tasks constructed in this way *mutually-exclusive*. For example, for N-way classification, each task consists of examples from N randomly sampled classes. The N classes are labeled from 1 to N, and critically, for each task, we *randomize* the assignment of classes to labels $\{1, 2, \ldots, N\}$. An visualization is provided in Appendix Figure D.1. In this illustration, the same class, such as the dog and butterfly, can be assigned different labels across tasks which makes it impossible for one model to solve all tasks simultaneously. This ensures that the task-specific class-to-label assignment cannot be inferred from a test input alone. However, the mutually-exclusive tasks requirement places a substantial burden on the user to cleverly design the meta-training setup (e.g., by shuffling labels or omitting goal information). While shuffling labels provides a reasonable mechanism to force tasks to be mutually-exclusive with standard few-shot image classification datasets such as MiniImageNet [114], this solution cannot be applied to all domains where we would like to utilize meta-learning. For example, consider meta-learning a pose predictor that can adapt to different objects: even if N different objects are used for meta-training, a powerful model can simply learn to ignore the training set for each task, and directly learn to predict the pose of each of the N objects. However, such a

model would not be able to adapt to *new* objects at meta-test time.

The primary contributions of this chapter are: 1) to identify and formalize the memorization problem in meta-learning, and 2) to propose an meta-regularizer (MR) using variational methods and information theory as a general approach for mitigating this problem *without* placing restrictions on the task distribution. The key insight of our variational meta-regularization approach is that the model acquired when memorizing tasks is more complex than the model that results from task-specific adaptation because the memorization model is a single model that simultaneously performs well on all tasks. It needs to contain all information in its weights needed to do well on test points without looking at training points. Therefore we would expect the information content of the weights of a memorization model to be larger, and hence the model should be more complex. As a result, we propose an objective that regularizes the information complexity of the meta-learned function class by variational method (motivated by Alemi et al. [4], Achille and Soatto [1]). Furthermore, we show that meta-regularization in MAML can be rigorously motivated by a PAC-Bayes bound on generalization. In a series of experiments on non-mutually-exclusive task distributions entailing both few-shot regression and classification, we find that memorization poses a significant challenge for both gradient-based [32] and contextual [36] meta-learning methods, resulting in near random performance on test tasks in some cases. Our meta-regularization approach enables both methods to achieve efficient adaptation and generalization, leading to substantial performance gains across

the board on non-mutually-exclusive tasks.

5.2 Preliminaries

We focus on the standard supervised meta-learning problem (see, e.g., Finn et al. [32]). As briefly stated in Chapter 1, we assume tasks $\mathcal{T}_i$ are sampled from a task distribution $p(\mathcal{T})$. During meta-training, for each task, we observe a set of training data $\mathcal{D}_i = (\boldsymbol{x}_i, \boldsymbol{y}_i)$ and a set of test data $\mathcal{D}_i^* = (\boldsymbol{x}_i^*, \boldsymbol{y}_i^*)$ with $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{iK})$,$\boldsymbol{y}_i = (y_{i1}, \ldots, y_{iK})$ sampled from $p(x, y|\mathcal{T}_i)$, and similarly for $\mathcal{D}_i^*$. We denote the entire meta-training set as $\mathcal{M} = \{\mathcal{D}_i, \mathcal{D}_i^*\}_{i=1}^N$. The goal of meta-training is to learn a model for a new task $\mathcal{T}$ by leveraging what is learned during meta-training and a small amount of training data for the new task $\mathcal{D}$. We use θ to denote the meta-parameters learned during meta-training and use ϕ to denote the task-specific parameters that are computed based on the task training data.

Following Grant et al. [44], Gordon et al. [43], given a meta-training set $\mathcal{M}$, we consider meta-learning algorithms that maximize conditional likelihood $q(\hat{y}^* = y^*|x^*, \theta, \mathcal{D})$, which is composed of three distributions: $q(\theta|\mathcal{M})$ that summarizes meta-training data into a distribution on meta-parameters, $q(\phi|\mathcal{D}, \theta)$ that summarizes the per-task training set into a distribution on task-specific parameters, and $q(\hat{y}^*|x^*, \phi, \theta)$ that is the predictive distribution. These distributions are learned to minimize

$$-\frac{1}{N}\sum_i \mathbb{E}_{q(\theta|\mathcal{M})q(\phi|\mathcal{D}_i,\theta)}\left[\frac{1}{K}\sum_{(x^*,y^*)\in\mathcal{D}_i^*}\log q(\hat{y}^* = y^*|x^*, \phi, \theta)\right]. \tag{5.1}$$

For example, in MAML [32], θ and ϕ are the weights of a predictor network, $q(\theta|\mathcal{M})$ is a delta function learned over the meta-training data, $q(\phi|\mathcal{D}, \theta)$ is a delta function centered at a point defined by gradient optimization, and ϕ parameterizes the predictor network $q(\hat{y}^*|x^*, \phi)$ [44]. In particular, to determine the task-specific parameters ϕ, the task training data $\mathcal{D}$ and θ are used in the predictor model $\phi = \theta + \frac{\alpha}{K} \sum_{(x,y)\in\mathcal{D}} \nabla_\theta \log q(y|x, \phi = \theta)$.

Another family of meta-learning algorithms are contextual methods [124], such as conditional neural processes (CNP) [36]. CNP instead defines $q(\phi|\mathcal{D}, \theta)$ as a mapping from $\mathcal{D}$ to a summary statistic ϕ (parameterized by θ). In particular, $\phi = a_\theta \circ h_\theta(\mathcal{D})$ is the output of an aggregator $a_\theta(\cdot)$ applied to features $h_\theta(\mathcal{D})$ extracted from the task training data. Then θ parameterizes a predictor network that takes ϕ and x^* as input and produces a predictive distribution $q(\hat{y}^*|x^*, \phi, \theta)$.

In the following sections, we describe a common pitfall for a variety of meta-learning algorithms, including MAML and CNP, and a general meta-regularization approach to prevent this pitfall.

5.3 The Memorization Problem in Meta-Learning

The ideal meta-learning algorithm will learn in such a way that generalizes to novel tasks. However, we find that unless tasks are carefully designed, current meta-learning algorithms can overfit to the tasks and end up ignoring the task training data (i.e., either $q(\phi|\mathcal{D}, \theta)$ does not depend on $\mathcal{D}$ or $q(\hat{y}^*|x^*, \phi, \theta)$ does not depend on ϕ, as shown in Figure 5.1), which can lead

to poor generalization. This memorization phenomenon is best understood through examples.

Consider a 3D object pose prediction problem (illustrated in Figure 5.1), where each object has a fixed canonical pose. The (x, y) pairs for the task are 2D grey-scale images of the rotated object (x) and the rotation angle relative to the fixed canonical pose for that object (y). In the most extreme case, for an unseen object, the task is impossible without using $\mathcal{D}$ because the canonical pose for the unseen object is unknown. The number of objects in the meta-training dataset is small, so it is straightforward for a single network to memorize the canonical pose for each training object and to infer the object from the input image (i.e., task overfitting), thus achieving a low training error without using $\mathcal{D}$. However, by construction, for a new object and canonical orientation, the task cannot be solved without using task training data to infer the canonical orientation. Therefore, this solution to memorize the canonical orientation of the meta-training objects will necessarily have poor generalization to test tasks with unseen objects.

As another example, imagine an automated medical prescription system that suggests medication prescriptions to doctors based on patient symptoms and the patient's previous record of prescription responses (i.e., medical history) for adaptation. In the meta-learning framework, each patient represents a separate task. Here, the symptoms and prescriptions have a close relationship, so we *cannot* assign random prescriptions to symptoms, in contrast to the classification tasks where we *can* randomly shuffle the labels to create mutually-

exclusiveness. For this non-mutually-exclusive task distribution, a standard meta-learning system can memorize the patients' identity information in the training, leading it to ignore the medical history and only utilize the symptoms combined with the memorized information. As a result, it may issue highly accurate prescriptions on the *meta-training* set, but fail to adapt to new patients effectively. While such a system would achieve a baseline level of accuracy for new patients, it would be no better than a standard supervised learning method applied to the pooled data. We formally define (complete) memorization as:

Definition 1 (Complete Meta-Learning Memorization)**.** *Complete memorization in meta-learning is when the learned model ignores the task training data such that* $I(\hat{y}^*; \mathcal{D}|x^*, \theta) = 0$ *(i.e.,* $q(\hat{y}^*|x^*, \theta, \mathcal{D}) = q(\hat{y}^*|x^*, \theta) = \mathbb{E}_{\mathcal{D}'|x^*}\left[q(\hat{y}^*|x^*, \theta, \mathcal{D}')\right]$*).*

Memorization describes an issue with overfitting the meta-training tasks, but it does not preclude the network from generalizing to unseen (x, y) pairs on the tasks similar to the training tasks. Memorization becomes an undesired problem for generalization to new tasks when $I(y^*; \mathcal{D}|x^*) \gg I(\hat{y}^*; \mathcal{D}|x^*, \theta)$ (i.e., the task training data is necessary to achieve good performance, even with exact inference under the data generating distribution, to make accurate predictions).

A model with the memorization problem may generalize to new datapoints in training tasks but cannot generalize to novel tasks, which distinguishes it from typical overfitting in supervised learning. In practice, we find that MAML and CNP frequently converge to this memorization solution (Table 5.2).

For MAML, memorization can occur when a particular setting of θ that does not adapt to the task training data can achieve comparable meta-training error to a solution that adapts θ. For example, if a setting of θ can solve all of the meta-training tasks (i.e., for all (x, y) in $\mathcal{D}$ and $\mathcal{D}^*$ the predictive error is close to zero), the optimization may converge to a stationary point of the MAML objective where minimal adaptation occurs based on the task training set (i.e., $\phi \approx \theta$). For a novel task where it is necessary to use the task training data, MAML can in principle still leverage the task training data because the adaptation step is based on gradient descent. However, in practice, the poor initialization of θ can affect the model's ability to generalize from a small mount of data. For CNP, memorization can occur when the predictive distribution network $q(\hat{y}^*|x^*, \phi, \theta)$ can achieve low training error without using the task training summary statistics ϕ. On a novel task, the network is not trained to use ϕ, so it is unable to use the information extracted from the task training set to effectively generalize.

In some problem domains, the memorization problem can be avoided by carefully constructing the tasks. For example, for N-way classification, each task consists of examples from N randomly sampled classes. If the classes are assigned to a random permutation of N for each task, this ensures that the task-specific class-to-label assignment cannot be inferred from the test inputs alone. As a result, a model that ignores the task training data cannot achieve low training error, preventing convergence to the memorization problem. We refer to tasks constructed in this way as *mutually-exclusive*. However, the

mutually-exclusive tasks requirement places a substantial burden on the user to cleverly design the meta-training setup (e.g., by shuffling labels or omitting goal information) and cannot be applied to all domains where we would like to utilize meta-learning.

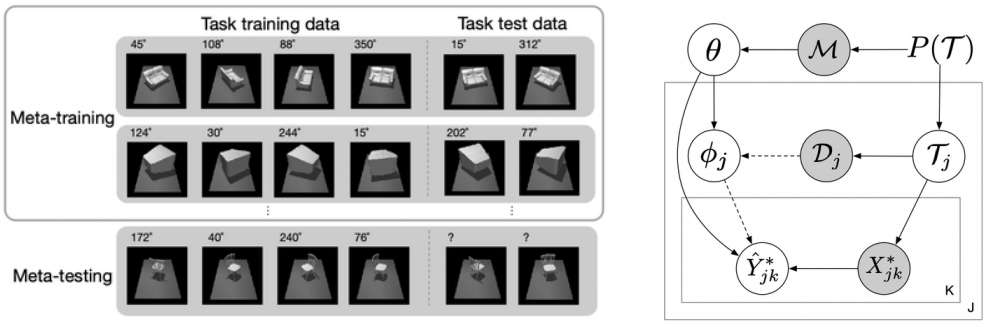


Figure 5.1: Left: An example of non-mutually-exclusive pose prediction tasks, which may lead to the memorization problem. Right: Graphical model for meta-learning. Observed variables are shaded. The complete memorization is the case without either one of the dashed arrows.

5.4 Meta Regularization Using Variational Methods

At a high level, the sources of information in the predictive distribution $q(\hat{y}^*|x^*, \theta, \mathcal{D})$ come from the input, the meta-parameters, and the data. The memorization problem occurs when the model encodes task information in the predictive network that is readily available from the task training set (i.e., it memorizes the task information for each meta-training task). We could resolve this problem by encouraging the model to minimize the training error and to rely on the task training dataset as much as possible for the prediction of y^* (i.e., to maximize $I(\hat{y}^*; \mathcal{D}|x^*, \theta)$). Explicitly maximizing $I(\hat{y}^*; \mathcal{D}|x^*, \theta)$ requires

an intractable marginalization over task training sets to compute $q(\hat{y}^*|x^*, \theta)$. Instead, we can implicitly encourage it by restricting the information flow from other sources (x^* and θ) to $\hat{y}^*$. To achieve both low error and low mutual information between $\hat{y}^*$ and (x^*, θ), the model must use task training data $\mathcal{D}$ to make predictions, hence increasing the mutual information $I(\hat{y}^*; \mathcal{D}|x^*, \theta)$, leading to reduced memorization. In this section, we describe two tractable ways to achieve this.

5.4.1 Meta Regularization on Activations

Given θ, the dependency between x^* and $\hat{y}^*$ is controlled by the direct path from x^* to $\hat{y}^*$ and the indirect path through $\mathcal{D}$ (see Figure 5.1), where the latter is desirable because it leverages the task training data. We can control the information flow between x^* and $\hat{y}^*$ by introducing an intermediate stochastic bottleneck variable z^* such that $q(\hat{y}^*|x^*, \phi, \theta) = \int q(\hat{y}^*|z^*, \phi, \theta) q(z^*|x^*, \theta)\, dz^*$ [4] as shown in Figure 5.2. Now, we would like to maximize $I(\hat{y}^*; \mathcal{D}|z^*, \theta)$ to prevent memorization. It can be bounded by

$$
\begin{aligned}
I(\hat{y}^*; \mathcal{D}|z^*, \theta) \geq & I(x^*; \hat{y}^*|\theta, z^*) \\
= & I(x^*; \hat{y}^*|\theta) - I(x^*; z^*|\theta) + I(x^*; z^*|\hat{y}^*, \theta) \\
\geq & I(x^*; \hat{y}^*|\theta) - I(x^*; z^*|\theta) \\
= & I(x^*; \hat{y}^*|\theta) - \mathbb{E}_{p(x^*)q(z^*|x^*,\theta)} \left[\log \frac{q(z^*|x^*, \theta)}{q(z^*|\theta)} \right] \\
\geq & I(x^*; \hat{y}^*|\theta) - \mathbb{E} \left[\log \frac{q(z^*|x^*, \theta)}{r(z^*)} \right] \\
= & I(x^*; \hat{y}^*|\theta) - \mathbb{E} \left[\mathcal{D}_{\mathrm{KL}}(q(z^*|x^*, \theta)||r(z^*)) \right] \qquad (5.2)
\end{aligned}
$$

where $r(z^*)$ is a variational approximation to the marginal, the first inequality follows from the statistical dependencies in our model (see Figure 5.2 and Appendix D.2 for the proof). By simultaneously minimizing $\mathbb{E}\left[\mathcal{D}_{\mathrm{KL}}(q(z^*|x^*,\theta)||r(z^*))\right]$ and maximizing the mutual information $I(x^*;\hat{y}^*|\theta)$, we can implicitly encourage the model to use the task training data $\mathcal{D}$.

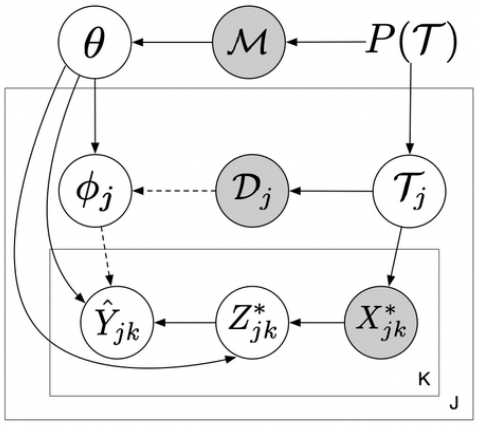


Figure 5.2: Graphical model of the regularization on activations. Observed variables are shaded and Z is bottleneck variable. The complete memorization corresponds to the graph without the dashed arrows.

For non-mutually-exclusive problems, the true label y^* is dependent on x^*. Marginalizing out x^* and $\mathcal{D}$, the distribution $q(\hat{y}^*|\theta)$ is spread out over all possible labels. If the model has memorization problem and $I(x^*;\hat{y}^*|\theta) = 0$, we have $q(\hat{y}^*|x^*,\theta,\mathcal{D}) = q(\hat{y}^*|x^*,\theta) = q(\hat{y}^*|\theta)$. Hence the prediction generated from $q(\hat{y}^*|x^*,\theta,\mathcal{D})$ has low accuracy. This suggests minimizing the training loss in Eq. (5.1) can increase $I(\hat{y}^*;\mathcal{D}|x^*,\theta)$ or $I(x^*;\hat{y}^*|\theta)$. Replacing the maximization of $I(x^*;\hat{y}^*|\theta)$ in Eq. (5.2) with minimizing the training loss results in the

following regularized training objective

$$\frac{1}{N}\sum_i \mathbb{E}_{q(\theta|\mathcal{M})q(\phi|\mathcal{D}_i,\theta)}\left[-\frac{1}{K}\sum_{(x^*,y^*)\in\mathcal{D}_i^*}\log q(\hat{y}^*=y^*|x^*,\phi,\theta)+\beta\mathcal{D}_{\mathrm{KL}}(q(z^*|x^*,\theta)||r(z^*))\right] \tag{5.3}$$

where $\log q(\hat{y}^*|x^*,\phi,\theta)$ is estimated by $\log q(\hat{y}^*|z^*,\phi,\theta)$ with $z^*\sim q(z^*|x^*,\theta)$, β modulates the regularizer and we set $r(z^*)$ as $\mathcal{N}(z^*;0,I)$. We refer to this regularizer as meta-regularization (MR) on the activations.

As we demonstrate in Section 5.6, we find that this regularizer performs well, but in some cases can fail to prevent the memorization problem. Our hypothesis is that in these cases, the network can sidestep the information constraint by storing the prediction of y^* in a part of z^*, which incurs only a small penalty in Eq. (5.3) and small lower bound in Eq. (5.2)

5.4.2 Meta Regularization on Weights

Alternatively, we can penalize the task information stored in the meta-parameters θ. Here, we provide an informal argument and provide the complete argument in Appendix D.3. Analogous to the supervised setting [1], given meta-training dataset $\mathcal{M}$, we consider θ as random variable where the randomness can be introduced by training stochasticity. We model the stochasticity over θ with a Gaussian distribution $\mathcal{N}(\theta;\theta_\mu,\theta_\sigma)$ with learned mean and variance parameters per dimension [18, 1]. By penalizing $I(y^*_{1:N},\mathcal{D}_{1:N};\theta|x^*_{1:N})$, we can limit the information about the training tasks stored in the meta-parameters θ and thus require the network to use the task training data to make accurate

predictions. We can tractably upper bound it by

$$I(y^*_{1:N}, \mathcal{D}_{1:N}; \theta | x^*_{1:N}) = \mathbb{E}\big[\log \tfrac{q(\theta|\mathcal{M})}{q(\theta|x^*_{1:N})}\big] \leq \mathbb{E}\left[\mathcal{D}_{\mathrm{KL}}\left(q(\theta|\mathcal{M}) \| r(\theta)\right)\right],$$

where $r(\theta)$ is a variational approximation to the marginal, which we set to $\mathcal{N}(\theta; 0, I)$. In practice, we apply meta-regularization to the meta-parameters θ that are not used to adapt to the task training data and denote the other parameters as $\tilde{\theta}$. In this way, we control the complexity of the network that can predict the test labels without using task training data, but we do not limit the complexity of the network that processes the task training data. Our final meta-regularized objective can be written as

$$\tfrac{1}{N}\textstyle\sum_i \mathbb{E}_{q(\theta;\theta_\mu,\theta_\sigma)q(\phi|\mathcal{D}_i,\tilde{\theta})}\left[-\tfrac{1}{K}\sum_{(x^*,y^*)\in\mathcal{D}^*_i} \log q(\hat{y}^* = y^*|x^*, \phi, \theta, \tilde{\theta}) + \beta\mathcal{D}_{\mathrm{KL}}(q(\theta;\theta_\mu,\theta_\sigma)||r(\theta))\right] \tag{5.4}$$

For MAML, we apply meta-regularization to the parameters uninvolved in the task adaptation. For CNP, we apply meta-regularization to the encoder parameters. The detailed algorithms are shown in Algorithm 5 and 6 in the appendix.

5.4.3 Does Meta Regularization Lead to Better Generalization?

Now that we have derived meta regularization approaches for mitigating the memorization problem, we theoretically analyze whether meta regularization leads to better generalization via a PAC-Bayes bound. In particular, we study meta regularization (MR) on the weights (W) of MAML, i.e. MR-MAML (W), as a case study.

Meta regularization on the weights of MAML uses a Gaussian distribution $\mathcal{N}(\theta; \theta_\mu, \theta_\sigma)$ to model the stochasticity in the weights. Given a task and task training data, the expected error is given by

$$er(\theta_\mu, \theta_\sigma, \mathcal{D}, \mathcal{T}) = \mathbb{E}_{\theta \sim \mathcal{N}(\theta; \theta_\mu, \theta_\sigma), \phi \sim q(\phi|\theta, \mathcal{D}), (x^*, y^*) \sim p(x, y|\mathcal{T})} \left[\mathcal{L}(x^*, y^*, \phi) \right], \tag{5.5}$$

where the prediction loss $\mathcal{L}(x^*, y^*, \phi_i)$ is bounded[1]. Then, we would like to minimize the error on novel tasks

$$er(\theta_\mu, \theta_\sigma) = \mathbb{E}_{\mathcal{T} \sim p(\mathcal{T}), \mathcal{D} \sim p(x, y|\mathcal{T})} \left[er(\theta_\mu, \theta_\sigma, \mathcal{D}, \mathcal{T}) \right] \tag{5.6}$$

We only have a finite sample of training tasks, so computing $er(Q)$ is intractable, but we can form an empirical estimate:

$$\begin{aligned} &\hat{er}(\theta_\mu, \theta_\sigma, \mathcal{D}_1, \mathcal{D}_1^*, ..., \mathcal{D}_n, \mathcal{D}_n^*) \\ =&\frac{1}{n} \sum_{i=1}^{n} \underbrace{\mathbb{E}_{\theta \sim \mathcal{N}(\theta; \theta_\mu, \theta_\sigma), \phi_i \sim q(\phi|\theta, \mathcal{D}_i)} \left[-\frac{1}{K} \sum_{(x^*, y^*) \in \mathcal{D}_i^*} \log q(\hat{y}^* = y^* | x^*, \phi_i) \right]}_{\hat{er}(\theta_\mu, \theta_\sigma, \mathcal{D}_i, \mathcal{D}_i^*)} \end{aligned} \tag{5.7}$$

where for exposition we have assumed $|\mathcal{D}_i^*| = K$ are the same for all tasks. We would like to relate $er(\theta_\mu, \theta_\sigma)$ and $\hat{er}(\theta_\mu, \theta_\sigma, \mathcal{D}_1, \mathcal{D}_1^*, ..., \mathcal{D}_n, \mathcal{D}_n^*)$, but the challenge is that θ_μ and θ_σ are derived from the meta-training tasks $\mathcal{D}_1, \mathcal{D}_1^*$, $\cdots$, $\mathcal{D}_n, \mathcal{D}_n^*$. There are two sources of generalization error: (i) error due to the finite number of observed tasks and (ii) error due to the finite number of examples observed per task. Closely following the arguments in [7], we apply a

[1]In practice, $\mathcal{L}(x^*, y^*, \phi_i)$ is MSE on a bounded target space or classification accuracy. We optimize the negative log-likelihood as a bound on the 0-1 loss.

standard PAC-Bayes bound to each of these and combine the results with a union bound, resulting in the following Theorem.

Theorem 5. *Let $P(\theta)$ be an arbitrary prior distribution over θ that does not depend on the meta-training data. Then for any $\delta \in (0, 1]$, with probability at least $1 - \delta$, the following inequality holds uniformly for all choices of θ_μ and θ_σ,*

$$er(\theta_\mu, \theta_\sigma) \leq \frac{1}{n} \sum_{i=1}^{n} \hat{er}(\theta_\mu, \theta_\sigma, \mathcal{D}_i, \mathcal{D}_i^*) + \left(\sqrt{\frac{1}{2(K-1)}} + \sqrt{\frac{1}{2(n-1)}} \right) \sqrt{\mathcal{D}_{KL}(\mathcal{N}(\theta; \theta_\mu, \theta_\sigma) \| P) + \log \frac{n(K+1)}{\delta}}, \tag{5.8}$$

where n is the number of meta-training tasks and K is the number of per-task validation datapoints.

We defer the proof to the Appendix D.4. The key difference from the result in [7] is that we leverage the fact that the task training data is split into training and validation.

In practice, we set $P(\theta) = r(\theta) = \mathcal{N}(\theta; 0, I)$. If we can achieve a low value for the bound, then with high probability, our test error will also be low. As shown in the Appendix D.4, by a first order Taylor expansion of the the second term of the RHS in Eq.(A.1) and setting the coefficient of the KL term as $\beta = \frac{\sqrt{1/2(K-1)} + \sqrt{1/2(n-1)}}{2\sqrt{\log n(K+1)/\delta}}$, we recover the MR-MAML(W) objective (Eq.(5.4)). β trades-off between the tightness of the generalization bound and the probability that it holds true. The result of this bound suggests that the proposed meta-regularization on weights does indeed improve generalization on the meta-test set.

5.5 Prior Work on Meta-Overfitting

Previous works have developed approaches for mitigating various forms of overfitting in meta-learning. These approaches aim to improve generalization in several ways: by reducing the number of parameters that are adapted in MAML [172], by compressing the task embedding [78], through data augmentation from a GAN [170], by using an auxiliary objective on task gradients [49], and via an entropy regularization objective [63]. These methods all focus on the setting with mutually-exclusive task distributions. We instead recognize and formalize the memorization problem, a particular form of overfitting that manifests itself with non-mutually-exclusive tasks, and offer a general and principled solution. Unlike prior methods, our approach is applicable to both contextual and gradient-based meta-learning methods. We additionally validate that prior regularization approaches, namely TAML [63], are not effective for addressing this problem setting.

Our derivation uses a Bayesian interpretation of meta-learning [138, 31, 28, 44, 43, 33, 67, 53]. Some Bayesian meta-learning approaches place a distributional loss on the inferred task variables to constrain them to a prior distribution [43, 111], which amounts to an information bottleneck on the latent *task variables*. Similarly Zintgraf et al. [172], Lee et al. [78], Guiroy et al. [49] aim to produce simpler or more compressed task adaptation processes. Our approach does the opposite, penalizing information from the *inputs* and *parameters*, to encourage the task-specific variables to contain greater information driven by the per-task data.

We use PAC-Bayes theory to study the generalization error of meta-learning and meta-regularization. Pentina and Lampert [107] extends the single task PAC-Bayes bound [86] to the multi-task setting, which quantifies the gap between empirical error on training tasks and the expected error on new tasks. More recent research shows that, with tightened generalization bounds as the training objective, the algorithms can reduce the test error for mutually-exclusive tasks [35, 7]. Our analysis is different from these prior works in that we only include pre-update meta parameters in the generalization bound rather than both pre-update and post-update parameters. In the derivation, we also explicitly consider the splitting of data into the task training set and task validation set, which is aligned with the practical setting.

The memorization problem differs from overfitting in conventional supervised learning in several aspects. First, memorization occurs at the task level rather than datapoint level and the model memorizes functions rather than labels. In particular, within a training task, the model can generalize to new datapoints, but it fails to generalize to new tasks. Second, the source of information for achieving generalization is different. For meta-learning the information is from both the meta-training data and new task training data but in standard supervised setting the information is only from training data. Finally, the aim of regularization is different. In the conventional supervised setting, regularization methods such as weight decay [72], dropout [133], the information bottleneck [141, 140], and Bayes-by-Backprop [18] are used to balance the network complexity and the information in the data. The aim of

meta-regularization is different. It governs the model complexity to avoid one complex model solving all tasks, while allowing the model's dependency on the task data to be complex. We further empirically validate this difference, finding that standard regularization do not solve the memorization problem.

5.6 Experimental Results

In the experimental evaluation, we aim to answer the following questions: (1) How prevalent is the memorization problem across different algorithms and domains? (2) How does the memorization problem affect the performance of algorithms on non-mutually-exclusive task distributions? (3) Is our meta-regularization approach effective for mitigating the problem and is it compatible with multiple types of meta-learning algorithms? (4) Is the problem of memorization empirically distinct from that of the standard overfitting problem?

To answer these questions, we propose several meta-learning problems involving non-mutually-exclusive task distributions, including two problems that are adapted from prior benchmarks with mutually-exclusive task distributions. We consider model-agnostic meta-learning (MAML) and conditional neural processes (CNP) as representative meta-learning algorithms. We study both variants of our method in combination with MAML and CNP. When comparing with meta-learning algorithms with and without meta-regularization, we use the same neural network architecture, while other hyperparameters are tuned via cross-validation per-problem.

5.6.1 Sinusoid Regression

First, we consider a toy sinusoid regression problem that is non-mutually-exclusive. The data for each task is created in the following way: the amplitude A of the sinusoid is uniformly sampled from a set of 20 equally-spaced points $\{0.1, 0.3, \cdots, 4\}$; u is sampled uniformly from $[-5, 5]$ and y is sampled from $\mathcal{N}(A\sin(u), 0.1^2)$. We provide both u and the amplitude A (as a one-hot vector) as input, i.e. $x = (u, A)$.

Table 5.1: Test MSE for the non-mutually-exclusive sinusoid regression problem.

Methods	MAML	MR-MAML (A) (ours)	MR-MAML (W) (ours)	CNP	MR-CNP (A) (ours)	MR-CNP (W) (ours)
5 shot	0.46 (0.04)	**0.17 (0.03)**	**0.16 (0.04)**	0.91 (0.10)	**0.10 (0.01)**	**0.11 (0.02)**
10 shot	0.13 (0.01)	**0.07 (0.02)**	**0.06 (0.01)**	0.92 (0.05)	**0.09 (0.01)**	**0.09 (0.01)**

At the test time, we expand the range of the tasks by randomly sampling the data-generating amplitude A uniformly from $[0.1, 4]$ and use a random one-hot vector for the input to the network. The meta-training tasks are a proper subset of the meta-test tasks.

In Table 5.1, we compare MAML and CNP against meta-regularized MAML (MR-MAML) and meta-regularized CNP (MR-CNP) where regularization is either on the activations (A) or the weights (W). We report the mean over 5 trials and the standard deviation in parentheses. Without the additional amplitude input, both MAML and CNP can easily solve the task and generalize to the meta-test tasks. However, once we add the additional amplitude input which indicates the task identity, we find that both MAML and CNP converge to the complete memorization solution and fail to generalize well to test data

(Appendix Figures D.2 and D.3). Both meta-regularized MAML and CNP (MR-MAML) and (MR-CNP) instead converge to a solution that adapts to the data, and as a result, greatly outperform the unregularized methods.

As shown in Figures 5.3, D.2 and D.3, when meta-learning algorithms converge to the memorization solution, the test tasks must be similar to the train tasks in order to achieve low test error. For CNP, although the task training set contains sufficient information to infer the correct amplitude, this information is ignored and the regression curve at test-time is determined by the one-hot vector. As a result, CNP can only generalize to points from the curves it has seen in the training (Figure D.2 first row). On the other hand, MAML does use the task training data (Figure 5.3, D.3 and Table 5.1), however, its performance is much worse than in the mutually-exclusive task. MR-MAML and MR-CNP avoid converging to a memorization solution and achieve excellent test performance on sinusoid task.

To illustrate the intuition that the model acquired when memorizing tasks is more complex than the model that results from task-specific adaptation, we plot the weight matrix for both MAML and CNP, with or without meta-regularization on the weights. The input $x = (u, A)$ where $u \sim \text{Unif}(-5, 5)$, A is 20 dimensional one-hot vector and the intermediate layer is 100 dimensional, hence $x \in \mathbb{R}^{21}$ and $W \in \mathbb{R}^{21\times 100}$. For both CNP and MAML, the meta-regularization restricts the part of weights that is connected to A close to 0. Therefore it avoids storing the amplitude information in weights and forces the amplitude to be inferred from the task training data $\mathcal{D}$, hence preventing the

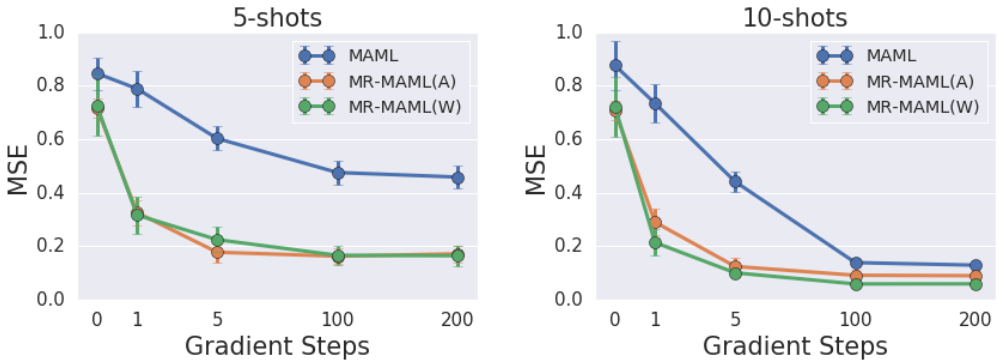


Figure 5.3: Test MSE on the mutually-non-exclusive sinusoid problem as function of the number of gradient steps used in the inner loop of MAML and MR-MAML. Each trial calculates the mean MSE over 100 randomly generated meta-testing tasks. The mean and standard deviation over 5 random trials are reported.

memorization problem.

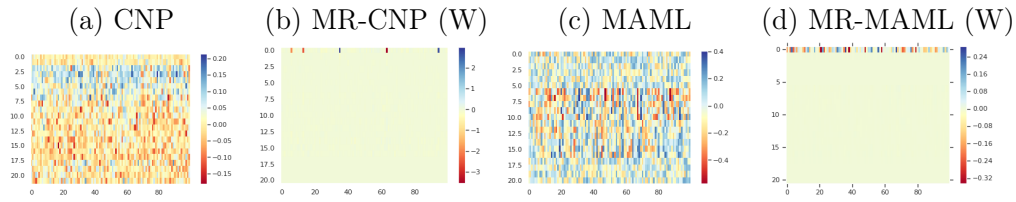


Figure 5.4: Visualization of the optimized weight matrix W that is connected to the inputs in the sinusoid regression example.

5.6.2 Pose Prediction

To illustrate the memorization problem on a more realistic task, we create a multi-task regression dataset based on the Pascal 3D data [156] (See Appendix D.5.1 for a complete description). We randomly select 50 objects for meta-training and the other 15 objects for meta-testing. For each object, we

use MuJoCo [144] to render images with random orientations of the instance on a table, visualized in Figure 5.1. For the meta-learning algorithm, the observation (x) is the 128×128 gray-scale image and the label (y) is the orientation relative to a fixed canonical pose. Because the number of objects in the meta-training dataset is small, it is straightforward for a single network to memorize the canonical pose for each training object and to infer the orientation from the input image, thus achieving a low meta-training error without using $\mathcal{D}$. However, this solution performs poorly at the test time because it has not seen the novel objects and their canonical poses.

Optimization modes and hyperparameter sensitivity. We choose the learning rate from $\{0.0001, 0.0005, 0.001\}$ for each method, β from $\{10^{-6}, 10^{-5}, \cdots, 1\}$ for meta-regularization and report the results with the best hyperparameters (as measured on the meta-validation set) for each method. In this domain, we find that the convergence point of the meta-learning algorithm is determined by both the optimization landscape of the objective and the training dynamics, which vary due to stochastic gradients and the random initialization. In particular, we observe that there are two modes of the objective, one that corresponds to complete memorization and one that corresponds to successful adaptation to the task data. As illustrated in the Figure 5.5, we find that models that converge to a memorization solution have lower training error than solutions which use the task training data, indicating a clear need for meta-regularization. When the meta-regularization is on the activations, the solution that the algorithms converge to depends on the learning rate,

while MR on the weights consistently converges to the adaptation solution (See Appendix for the sensitivity analysis).

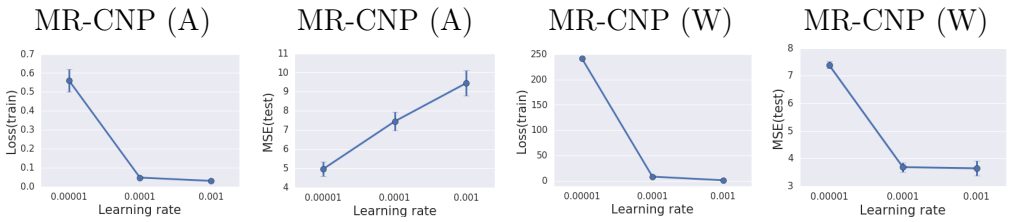


Figure 5.5: Sensitivity of activation regularization and weight regularization with respect to the learning rate on the pose prediction problem.

This suggests that MR on the activations is not always successful at preventing memorization. Our hypothesis is that there exists a solution in which the bottlenecked activations encode only the prediction y^*, and discard other information. Such a solution can achieve both low training MSE and low regularization loss without using task training data, particularly if the predicted label contains a small number of bits (i.e., because the *activations* will have low information complexity). However, note that this solution does not achieve low regularization error when applying MR to the weights because the *function* needed to produce the predicted label does not have low information complexity. As a result, meta-regularization on the weights does not suffer from this pathology and is robust to different learning rates. Therefore, we will use regularization on weights as the proposed methodology in the following experiments and algorithms in Appendix D.1.

Quantitative results. We compare MAML and CNP with their meta-regularized versions (Table 5.2). We report the average over 5 trials and

standard deviation in parentheses. We additionally include fine-tuning as baseline, which trains a single network on all the instances jointly, and then fine-tunes on the task training data. Meta-learning with meta-regularization (on weights) outperforms all competing methods by a large margin. We show test error as a function of the meta-regularization coefficient β in Figure 5.6. The curve reflects the trade-off when changing the amount of information contained in the weights. We observe β provides us a knob with which we can control the degree to which the algorithm adapts versus memorizes. When β is small, we observe memorization, leading to large test error; when β is too large, the network does not store enough information in the weights to perform the task. Crucially, in the middle of these two extremes, meta-regularization is effective in inducing adaptation, leading to good generalization. It gives a knob that allows us to tune the degree to which the model uses the data to adapt versus relying on the prior.

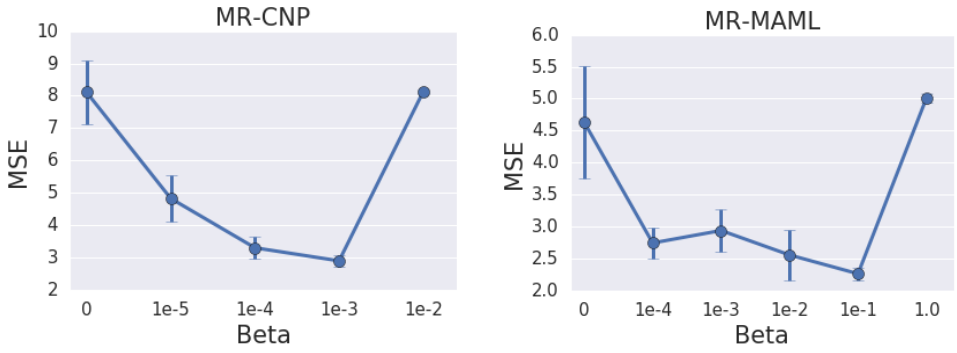


Figure 5.6: The performance of MAML and CNP with meta-regularization on the weights, as a function of the regularization strength β. The plot shows the mean and standard deviation across 5 meta-training runs.

Table 5.2: Meta-test MSE for the pose prediction problem. We compare MR-MAML (ours) with conventional MAML and fine-tuning (FT).

Method	MAML	MR-MAML (W) (ours)	CNP	MR-CNP (W) (ours)	FT	FT + Weight Decay
MSE	5.39 (1.31)	**2.26 (0.09)**	8.48 (0.12)	2.89 (0.18)	7.33 (0.35)	6.16 (0.12)

Comparison to standard regularization. We compare our meta regularization with standard regularization techniques, weight decay [72] and Bayes-by-Backprop [18], in Table 5.3. We report the mean and standard deviation over 5 random trials. We observe that simply applying standard regularization to all the weights, as in conventional supervised learning, does not solve the memorization problem, which validates that the memorization problem differs from the standard overfitting problem.

Table 5.3: Meta-test MSE for the pose prediction problem. We compare MR-CNP (ours) with conventional CNP, CNP with weight decay, and CNP with Bayes-by-Backprop (BbB) regularization on all the weights.

Methods	CNP	CNP + Weight Decay	CNP + BbB	MR-CNP (W) (ours)
MSE	8.48 (0.12)	6.86 (0.27)	7.73 (0.82)	**2.89 (0.18)**

5.6.3 Omniglot and MiniImagenet Classification

Next, we study memorization in the few-shot classification problem by adapting the few-shot Omniglot [76] and MiniImagenet [114, 150] benchmarks to the non-mutually-exclusive setting. In the *non-mutually-exclusive* N-way K-shot classification problem, each class is (randomly) assigned a fixed classification label from 1 to N. For each task, we randomly select a corresponding class for each classification label and K task training data points and K task test

data points from that class[2]. This ensures that each class takes only one classification label across tasks and different tasks are non-mutually-exclusive (See Appendix D.5.2 for details).

We evaluate MAML, TAML [63], MR-MAML (ours), fine-tuning, and a nearest neighbor baseline on non-mutually-exclusive classification tasks (Table 5.4). The fine-tuning and nearest-neighbor baseline results for MiniImagenet are from [114]. We find that MR-MAML significantly outperforms previous methods on all of these tasks. To better understand the problem, for the MAML variants, in Table 5.5 we report the pre-update accuracy for the non-mutually-exclusive classification experiment in Section 5.6.3. The pre-update accuracy is obtained by the initial parameters θ instead of the task adapted parameters ϕ.

Table 5.4: Meta-test accuracy on non-mutually-exclusive (NME) classification.

NME Omniglot	20-way 1-shot	20-way 5-shot
MAML	7.8 (0.2)%	50.7 (22.9)%
TAML [63]	9.6 (2.3)%	67.9 (2.3)%
MR-MAML (W)	**83.3 (0.8)**%	**94.1 (0.1)**%

NME MiniImagenet	5-way 1-shot	5-way 5-shot
Fine-tuning	28.9 (0.5))%	49.8 (0.8))%
Nearest-neighbor	41.1 (0.7)%	51.0 (0.7) %
MAML	26.3 (0.7)%	41.6 (2.6)%
TAML [63]	26.1 (0.6)%	44.2 (1.7)%
MR-MAML (W)	**43.6 (0.6)**%	**53.8 (0.9)**%

At the meta-training time, for both MAML and MR-MAML the post-update accuracy obtained by using ϕ gets close to 1. High pre-update accuracy reflects the memorization problem. For example, in 20-way 1-shot Omniglot

[2]We assume that the number of classes in the meta-training set is larger than N.

example, the pre-update accuracy for MAML is 99.2% at the training time, which means only 0.8% improvement in accuracy is due to adaptation, so the task training data is ignored to a large extent. The pre-update training accuracy for MR-MAML is 5%, which means 95% improvement in accuracy during training is due to the adaptation.

Table 5.5: Meta-training *pre-update* accuracy on non-mutually-exclusive classification.

NME Omniglot	20-way 1-shot	20-way 5-shot
MAML	99.2 (0.2)%	45.1 (38.9)%
TAML	68.9(43.1)%	6.7 (1.8)%
MR-MAML	**5.0 (0)**%	**5.0 (0)**%

NME MiniImagenet	5-way 1-shot	5-way 5-shot
MAML	99.4 (0.1)%	21.0(1.2)%
TAML	99.4 (0.1)%	20.8(0.4)%
MR-MAML	**20.0(0)**%	**20.2(0.1)**%

The high pre-update meta-training accuracy and low meta-test accuracy are evidence of the memorization problem for MAML and TAML, indicating that it is learning a model that ignores the task data. In contrast, MR-MAML successfully controls the pre-update accuracy to be near chance and encourages the learner to use the task training data to achieve low meta-training error, resulting in good performance at meta-test time.

Finally, we verify that meta-regularization does not degrade performance on the standard mutually-exclusive task. We evaluate performance as a function of regularization strength on the standard 20-way 1-shot Omniglot task in Figure 5.7, which shows the mean and standard deviation across 5 meta-training runs. We find that small values of β lead to slight improvements over MAML. This indicates that meta-regularization substantially improves performance in

the non-mutually-exclusive setting without degrading performance in other settings. Notice the accuracy numbers are not directly comparable to previous work (e.g., [32]) because we do not use data augmentation.

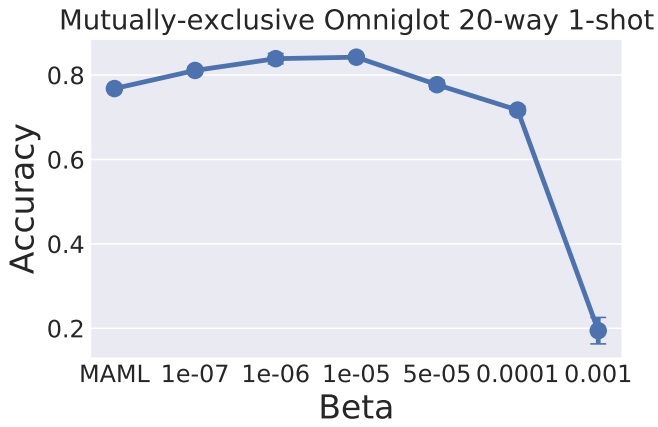


Figure 5.7: The test accuracy of MAML with meta-regularization on the weights as a function of the regularization strength β on the mutually-exclusive 20-way 1-shot Omniglot problem.

5.7 Conclusion and Discussion

Meta-learning has achieved remarkable success in few-shot learning problems. However, we identify a pitfall of current algorithms: the need to create task distributions that are mutually exclusive. This requirement restricts the domains that meta-learning can be applied to. We formalize the failure mode, i.e. the memorization problem, that results from training on non-mutually-exclusive tasks and distinguish it as a function-level overfitting problem compared to the the standard label-level overfitting in supervised learning.

We illustrate the memorization problem with different meta-learning algorithms on a number of domains. To address the problem, we propose an algorithm-agnostic meta-regularization (MR) approach that leverages an information-theoretic perspective of the problem. The key idea is that by placing a soft restriction on the information flow from meta-parameters in prediction of test set labels, we can encourage the meta-learner to use task training data during meta-training. We achieve this by successfully controlling the complexity of model prior to the task adaptation.

The memorization issue is quite broad and is likely to occur in a wide range of real-world applications, for example, personalized speech recognition systems, learning robots that can adapt to different environments [97], and learning goal-conditioned manipulation skills using trial-and-error data. Further, this challenge may also be prevalent in other conditional prediction problems, beyond meta-learning, an interesting direction for future study. By both recognizing the challenge of memorization and developing a general and lightweight approach for solving it, we believe that this work represents an important step towards making meta-learning algorithms applicable to and effective on any problem domain.

Chapter 6

Conclusion and Future Directions

In this thesis, we propose novel methodologies and theoretical analysis for variational methods. A common gist underlying different approaches is to utilize the dependence structures among random quantities. Modeling the dependence between random variables, we have proposed a framework that significantly ameliorates the uncertainty estimation of variational inference. We have proved the convergence properties of a pairwise dependent VI in a case study for community detection with SBM. Using the dependence between random samples, we have shown the efficiency of an unbiased, low-variance stochastic gradient estimation for discrete latent variables. Scrutinizing the relationship between correlated tasks, we have identified the memorization problem in meta-learning and proposed variational regularization as a solution.

There are active research works on variational methods. One future direction is to bridge the performance gap between VI and MCMC. In Chaper 2, we have shown the potential of VI to achieve accurate uncertainty estimation without sacrificing efficiency. Yet many open questions remain to be explored. For example, a theoretical understanding of the speed-accuracy tradeoff is essential to systematically choose hyper-parameters. A possibly more challenging

extension would be exploring alternative approaches to model the dependence structure, when the latent variables are extremely high-dimensional or accurate uncertainty estimation for a large number of local variables is needed.

A promising future direction is to construct the theoretical foundation for variational methods. There has been plenty of literature in the EM algorithm, which can shed light on the study of variational methods since both are iterative bound optimization algorithms. Recently, theoreticians have made progress in understanding MFVI but mainly restricted to specific models. Extracting similarities among such analysis to form a general theoretical framework for MFVI can be an influential direction. In practice, many studies have empirically suggested the benefits of structured variational inference. In Chaper 3, our preliminary analysis is in a simplified setting as a blockmodel with a pairwise structure and two equal-sized communities. A natural extension is generalizing to broad dependence structures, probabilistic models and related algorithms such as Belief Propagation.

In this thesis, the dependence structures are studied as statistical associations. An important future direction is to study the causal relationship. Causal inference in observational study estimates the treatment effects for the target population. The fundamental problem of causal inference is that observing all potential outcomes on a single unit is impossible. To ensure identifiability of causal effects, assumptions are necessary such as Stable Unit Treatment Value Assumption (SUTVA), positivity, consistency, and ignorability. Latent variable models and variational methods can facilitate causal reasoning

when certain assumptions are too strong to satisfy in practice. For example, ignorability assumes no unmeasured confounding. When this condition is not met, the unmeasured confounders can be modeled as latent variables. The latent variable models can probabilistically specify the dependence between observations and latent variables, between observed confounders and unmeasured confounders, while variational methods can be applied to make the inference.

Causal inference is related to the missing data problem. In this context, the counterfactual outcome can be viewed as the latent variable. Conditional on the observed data, the imputation of unobserved potential outcomes can be viewed as the posterior inference. The main challenge is that the data are missing structurally: all the potential outcomes without treatment are missing in the treatment group while all the potential outcomes with treatment are missing in the control group. In machine learning, such a prediction problem has been studied as the domain adaptation. Related latent variable models, such as the hierarchical Bayesian model and meta-learning, take advantage of globally shared and domain-specific variables for out-of-distribution generalization. Some encouraging preliminary results have suggested that latent variables and variational methods can be new tools for the counterfactual prediction.

Appendices

Appendix A

Appendix for Semi-Implicit Variational Inference

A.1 Proofs of Main Results

Proof of Inequility (2.3). To prove a functional form of Jensen's Inequality, let $h(\boldsymbol{z}) = \mathbb{E}_{\boldsymbol{\psi}\sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})} q(\boldsymbol{z}|\boldsymbol{\psi})$ and $\langle f, g\rangle_{L^2} = \int f(\boldsymbol{z})g(\boldsymbol{z})d\boldsymbol{z}$. From Theorem 1, we have convexity, and according to Theorem 6.2.1. of Kurdila and Zabarankin [74], we have an equivalent first-order definition for convexity as

$$\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z}|\boldsymbol{\psi})||p(\boldsymbol{z})) \geq \mathcal{D}_{\mathrm{KL}}(h(\boldsymbol{z})||p) + \langle q(\boldsymbol{z}|\boldsymbol{\psi}) - h(\boldsymbol{z}), \nabla_q \mathcal{D}_{\mathrm{KL}}(q||p)|_{h(\boldsymbol{z})}\rangle_{L^2}$$

Taking the expectation with respect to $\boldsymbol{\psi} \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$ on both sides, we have

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{\psi}\sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})} \mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z}|\boldsymbol{\psi})||p(\boldsymbol{z})) \\
\geq &\mathcal{D}_{\mathrm{KL}}(h(\boldsymbol{z})||p(\boldsymbol{z})) + \mathbb{E}_{\boldsymbol{\psi}\sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})}[\langle q(\boldsymbol{z}|\boldsymbol{\psi}) - h(\boldsymbol{z}), \nabla_q \mathcal{D}_{\mathrm{KL}}(q||p)|_{h(\boldsymbol{z})}\rangle_{L^2}] \\
= &\mathcal{D}_{\mathrm{KL}}(\mathbb{E}_{\boldsymbol{\psi}\sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})} q(\boldsymbol{z}|\boldsymbol{\psi})||p(\boldsymbol{z})).
\end{aligned}$$

□

Proof of Proposition 1. We show that directly maximizing the lower bound $\underline{\mathcal{L}}$ of ELBO in (2.4) may drive $q(\boldsymbol{\psi})$ towards degeneracy. For VI that uses

$q(\boldsymbol{z} \mid \boldsymbol{\psi})$ as its variational distribution, supposing $\boldsymbol{\psi}^*$ is the optimum variational parameter, which means

$$\boldsymbol{\psi}^* = \arg\max_{\boldsymbol{\psi}} -\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z} \mid \boldsymbol{\psi}) || p(\boldsymbol{x}, \boldsymbol{z})),$$

then we have

$$\begin{aligned}
\underline{\mathcal{L}} &= \int q_{\boldsymbol{\phi}}(\boldsymbol{\psi})[-\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z} \mid \boldsymbol{\psi}) || p(\boldsymbol{x}, \boldsymbol{z}))] d\boldsymbol{\psi} \\
&\leq \int q_{\boldsymbol{\phi}}(\boldsymbol{\psi}) d\boldsymbol{\psi}[-\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z} \mid \boldsymbol{\psi}^*) || p(\boldsymbol{x}, \boldsymbol{z}))] \\
&= -\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z} \mid \boldsymbol{\psi}^*) || p(\boldsymbol{x}, \boldsymbol{z})).
\end{aligned}$$

The equality in the above equation is reached if and only if $q(\boldsymbol{\psi}) = \delta_{\boldsymbol{\psi}^*}(\boldsymbol{\psi})$, which means the mixing distribution degenerates to a point mass density and hence SIVI degenerates to vanilla VI. □

Proof of Proposition 2. $I_0 = 0$ is trivial. Denote $\boldsymbol{\psi}^{(0)} = \psi_v$. For i.i.d. samples $\boldsymbol{\psi}^{(k)} \sim q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$, when $K \to \infty$, by the strong law of large numbers, $\tilde{h}_K(\boldsymbol{z}) = \frac{\sum_{k=0}^{K} q(\boldsymbol{z} \mid \boldsymbol{\psi}^{(k)})}{K+1}$ converges almost surely to $\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{\psi})} q(\boldsymbol{z} \mid \boldsymbol{\psi}) = h_{\boldsymbol{\phi}}(\boldsymbol{z})$. To prove (2.7), by the strong law of large numbers, we first rewrite it as the limit of a double sequence $S(K, J)$, where $K, J \in \{1, 2, \ldots, \}$, and check the condition for interchange of iterated limits [120, 50]: i) The double limit exists; ii) Fixing one index of the double sequence, for the other index the one side limit exists.

$$\begin{aligned}
&\lim_{K \to \infty} \mathbb{E}_{\boldsymbol{\psi}^{(0)}, \boldsymbol{\psi}^{(1)}, \cdots, \boldsymbol{\psi}^{(K)} \sim q(\boldsymbol{\psi})} \log \frac{\sum_{k=0}^{K} q(\boldsymbol{z} \mid \boldsymbol{\psi}^{(k)})}{K+1} \\
&= \lim_{K \to \infty} \lim_{J \to \infty} \frac{1}{J} \sum_{j=1}^{J} \log \frac{1}{K+1} \sum_{k=0}^{K} q(\boldsymbol{z} \mid \boldsymbol{\psi}_j^{(k)}) \\
&\triangleq \lim_{K \to \infty} \lim_{J \to \infty} S(K, J)
\end{aligned}$$

Here $\boldsymbol{\psi}_j^{(k)}$ are i.i.d. samples from $q(\boldsymbol{\psi})$. For i) we show double limit $\lim_{K,J\to\infty} S(K,J) = \log h(\boldsymbol{z})$. For $\forall\epsilon > 0$, $\exists N(\epsilon)$, when $K, J > N(\epsilon)$, $|\log \frac{1}{K+1}\sum_{k=0}^{K} q(\boldsymbol{z}\,|\,\boldsymbol{\psi}_j^{(k)}) - \log h(\boldsymbol{z})| < \epsilon$ thanks to the law of large numbers, then

$$\begin{aligned}&\left|\sum_{j=1}^{J}\log\frac{1}{K+1}\sum_{k=0}^{K} q(\boldsymbol{z}\,|\,\boldsymbol{\psi}_j^{(k)}) - J\log h(\boldsymbol{z})\right|\\ \leq&\sum_{j=1}^{J}\left|\log\frac{1}{K+1}\sum_{k=0}^{K} q(\boldsymbol{z}\,|\,\boldsymbol{\psi}_j^{(k)}) - \log h(\boldsymbol{z})\right| \leq J\epsilon\end{aligned}$$

Deviding both sides by J we get $|S(K,J) - \log h(\boldsymbol{z})| \leq \epsilon$ when $K, J > N(\epsilon)$. By definition, we have $\lim_{K,J\to\infty} S(K,J) = \log h(\boldsymbol{z})$.

ii) for each fixed $J \in \mathbb{N}$, $\lim_{K\to\infty} S(K,J) = \log h(\boldsymbol{z})$ exists; for each fixed $K \in \mathbb{N}$, $\lim_{J\to\infty} S(K,J) = \mathbb{E}_{\boldsymbol{\psi}^{(0)},\boldsymbol{\psi}^{(1)},\cdots,\boldsymbol{\psi}^{(K)}\sim q(\boldsymbol{\psi})} \log \frac{\sum_{k=0}^{K} q(\boldsymbol{z}\,|\,\boldsymbol{\psi}^{(k)})}{K+1} \leq \log h(\boldsymbol{z})$ also exists. The limitation can then be interchanged and (2.7) is proved. Therefore, we have

$$\begin{aligned}&\lim_{k\to\infty}\underline{\mathcal{L}}_k = \underline{\mathcal{L}} + \mathbb{E}_{\boldsymbol{\psi}}\mathcal{D}_{\mathrm{KL}}(q(\boldsymbol{z}\,|\,\boldsymbol{\psi})||h_{\boldsymbol{\phi}}(\boldsymbol{z}))\\ =&\mathbb{E}_{\boldsymbol{\psi}\sim q(\boldsymbol{\psi})}\mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z}\,|\,\boldsymbol{\psi})}\left[\log\frac{q(\boldsymbol{z}\,|\,\boldsymbol{\psi})}{h_{\boldsymbol{\phi}}(\boldsymbol{z})} - \log\frac{q(\boldsymbol{z}\,|\,\boldsymbol{\psi})}{p(x,\boldsymbol{z})}\right]\\ =&-\mathbb{E}_{\boldsymbol{\psi}\sim q(\boldsymbol{\psi})}\mathbb{E}_{\boldsymbol{z}\sim q(\boldsymbol{z}\,|\,\boldsymbol{\psi})}\log\frac{h_{\boldsymbol{\phi}}(\boldsymbol{z})}{p(x,\boldsymbol{z})}\\ =&-\mathcal{D}_{\mathrm{KL}}(h_{\boldsymbol{\phi}}(\boldsymbol{z})||p(x,\boldsymbol{z})) = \mathcal{L}\end{aligned}$$

□

A.2 Bayesian Logistic Regression

We consider datesets *waveform* ($n = 5000$, $V = 21$, and 400/4600 for training/testing), *spam* ($n = 3000$, $V = 2$, and 2000/1000 for training/testing),

and *nodal* ($n = 53$, $V = 5$, and 25/28 for training/testing). The training-set-size to feature-dimension ratio n_{train}/V clearly varies in these three datasets, and we expect the posterior uncertainty to be large if this ratio is small.

The contribution of observation i to the likelihood can be expressed as

$$P(y_i \,|\, \boldsymbol{x}_i, \boldsymbol{\beta}) = \frac{e^{y\boldsymbol{x}_i'\boldsymbol{\beta}}}{1 + e^{\boldsymbol{x}_i'\boldsymbol{\beta}}} \propto e^{(y-\frac{1}{2})\boldsymbol{x}_i'\boldsymbol{\beta}} \mathbb{E}_{\omega_i}\left[e^{-\frac{\omega_i(\boldsymbol{x}_i'\boldsymbol{\beta})^2}{2}}\right],$$

where the expectation is taken respect to $\omega_i \sim \text{PG}(1, 0)$, and hence we have an augmented likelihood as

$$P(y_i, \omega_i \,|\, \boldsymbol{x}_i, \boldsymbol{\beta}) \propto e^{(y-\frac{1}{2})\boldsymbol{x}_i'\boldsymbol{\beta} - \frac{1}{2}\omega_i(\boldsymbol{x}_i'\boldsymbol{\beta})^2}. \tag{A.1}$$

A.2.1 Gibbs Sampling via Data Augmentation

Denoting $\mathbf{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)'$, $\boldsymbol{y} = (y_1, \ldots, y_N)'$, $\mathbf{A} = \text{diag}(\alpha_0, \ldots, \alpha_V)'$, and $\boldsymbol{\Omega} = \text{diag}(\omega_1, \ldots, \omega_N)$, we have

$$(\omega_i \,|\, -) \sim \text{PG}(1, \boldsymbol{x}_i'\boldsymbol{\beta}), \quad (\boldsymbol{\beta} \,|\, -) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \tag{A.2}$$

where $\boldsymbol{\Sigma} = (\mathbf{A} + \mathbf{X}'\boldsymbol{\Omega}\mathbf{X})^{-1}$ and $\boldsymbol{\mu} = \boldsymbol{\Sigma}\mathbf{X}'(\boldsymbol{y} - 1/2)$.

A.2.2 Mean-Field Variational Inference with Diagonal Covariance Matrix

We choose a fully factorized Q distribution as

$$Q = \left[\prod\nolimits_i q(\omega_i)\right]\left[\prod\nolimits_v q(\beta_v)\right]. \tag{A.3}$$

To exploit conjugacy, defining

$$q(\omega_i) = \text{PG}\left(1, \lambda_i\right), \quad q(\beta_v) = \mathcal{N}(\mu_v, \sigma_v^2), \tag{A.4}$$

we have closed-form coordinate ascent variational inference update equations as

$$\lambda_i = \sqrt{\mathbb{E}[(\boldsymbol{x}_i'\boldsymbol{\beta})^2]} = \sqrt{\boldsymbol{x}_i'\mathbb{E}[\boldsymbol{\beta}\boldsymbol{\beta}']\boldsymbol{x}_i}, \quad \sigma_v^2 = \left(\mathbb{E}[\alpha_v] + \sum\nolimits_i \mathbb{E}[\omega_i] x_{iv}^2\right)^{-1}$$
$$\mu_v = \sigma_v^2 \sum\nolimits_i x_{iv} \left\{ y_i - 1/2 - \mathbb{E}[\omega_i] \sum\nolimits_{\tilde{v} \neq v} x_{i\tilde{v}} \mathbb{E}[\beta_{\tilde{v}}] \right\}, \tag{A.5}$$

where the expectations with respect to the q distributions can be expressed as $\mathbb{E}[\boldsymbol{\beta}\boldsymbol{\beta}'] = \boldsymbol{\mu}\boldsymbol{\mu}' + \text{diag}(\sigma_0^2, \ldots, \sigma_V^2)$ and $\mathbb{E}[\omega_i] = \tanh(\lambda_i/2)/(2\lambda_i)$.

A.2.3 Mean-Field Variational Inference with Full Covariance Matrix

We choose a fully factorized Q distribution as

$$Q = \left[\prod\nolimits_i q(\omega_i)\right] q(\boldsymbol{\beta}), q(\omega_i) = \text{PG}\left(1, \lambda_i\right), q(\boldsymbol{\beta}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}). \tag{A.6}$$

We have closed-form coordinate ascent variational inference update equations as

$$\lambda_i = \sqrt{\mathbb{E}[(\boldsymbol{x}_i'\boldsymbol{\beta})^2]} = \sqrt{\boldsymbol{x}_i'\mathbb{E}[\boldsymbol{\beta}\boldsymbol{\beta}']\boldsymbol{x}_i}, \quad \boldsymbol{\Sigma} = (\mathbb{E}[\mathbf{A}] + \mathbf{X}'\mathbb{E}[\boldsymbol{\Omega}]\mathbf{X})^{-1}, \ \boldsymbol{\mu} = \boldsymbol{\Sigma}\mathbf{X}'(\boldsymbol{y} - 1/2),$$

where the expectations with respect to the q distributions can be expressed as $\mathbb{E}[\boldsymbol{\beta}\boldsymbol{\beta}'] = \boldsymbol{\mu}\boldsymbol{\mu}' + \boldsymbol{\Sigma}$ and $\mathbb{E}[\omega_i] = \tanh(\lambda_i/2)/(2\lambda_i)$.

A.2.4 SIVI Configuration

For inputs in Algorithm 1, we choose multi-layer perceptron with layer size $[100, 200, 100]$ as $T_{\boldsymbol{\phi}}$ for $\boldsymbol{\psi} = T_{\boldsymbol{\phi}}(\boldsymbol{\epsilon})$, $\boldsymbol{\epsilon}$ as 50 dimensional isotropic Gaussian random variable and $K = 1000$, $J = 50$. We choose multivariate normal as

explicit distribution $q_{\boldsymbol{\xi}}(\boldsymbol{z} \mid \boldsymbol{\psi}) = \mathcal{N}(\boldsymbol{z}; \boldsymbol{\psi}, \boldsymbol{\xi})$. In this setting, $\boldsymbol{\psi}$ is the mean variable generated from $q_{\boldsymbol{\phi}}(\boldsymbol{\psi})$ while $\boldsymbol{\xi}$ is the covariance matrix which can be either diagonal or full. In the experiments, we update the neural network parameter $\boldsymbol{\phi}$ by Adam optimizer and update $\boldsymbol{\xi}$ by gradient descent. The implicit layer parameter $\boldsymbol{\phi}$ and explicit layer parameter $\boldsymbol{\xi}$ are updated iteratively.

A.3 Experimental Settings and Results for SIVAE

We implement SIVI with $M = 3$ stochastic hidden layers, with the dimensions of hidden layers $[\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \boldsymbol{\ell}_3]$ as $[150, 150, 150]$ and with the dimensions of injected noises $[\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2, \boldsymbol{\epsilon}_3]$ as $[150, 100, 50]$. Between two adjacent stochastic layers there is a fully connected deterministic layer with size 500 and $ReLU$ activation function. We choose binary pepper and salt noise [60] for $q_t(\boldsymbol{\epsilon})$. The model is trained for 2000 epochs with mini-batch size 200 and step-size $0.001 * 0.75^{\text{epoch}/100}$. K_t is gradually increased from 1 to 100 during the first 1500 epochs. The explicit and implicit layers are trained iteratively. Warm-up is used during the first 300 epochs as suggested by Sønderby et al. [132] to gradually impose the prior regularization term $\mathcal{D}_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) || p(\boldsymbol{z}))$. The model is trained end-to-end using the Adam optimizer. After training process, as in Rezende et al. [117] and Burda et al. [19], we compute the marginal likelihood for test set by importance sampling with $S = 2000$:

$$\log p(\boldsymbol{x}) \approx \log \frac{1}{S} \sum_{s=1}^{S} \frac{p(\boldsymbol{x} \mid \boldsymbol{z}_s) p(\boldsymbol{z}_s)}{h(\boldsymbol{z}_s|\boldsymbol{x})}, \quad \boldsymbol{z}_s \sim h(\boldsymbol{z}_s|\boldsymbol{x}).$$

A.4 Additional Figures

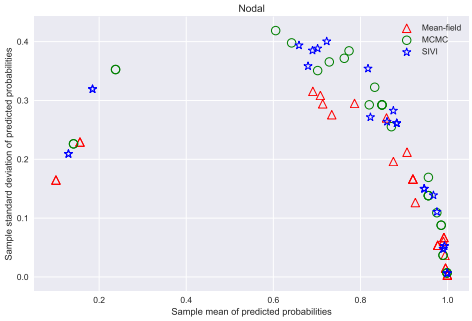


Figure A.1: Sample means and standard deviations of predictive probabilities for dataset *nodal*.

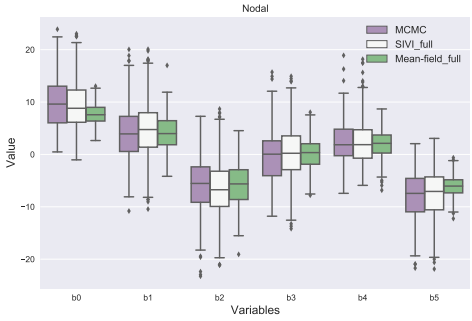


Figure A.2: Boxplot of marginal posteriors inferred by MCMC, SIVI, and MFVI for dataset *nodal*.

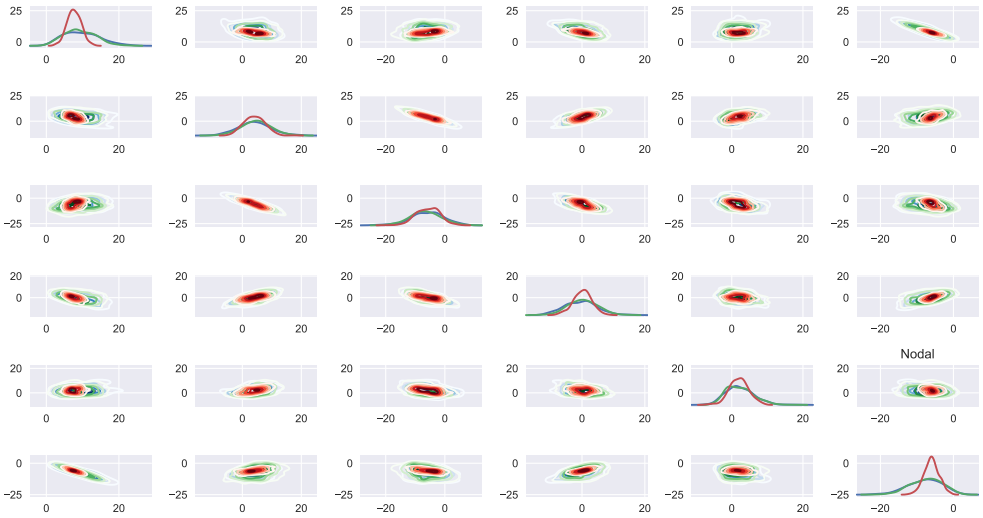


Figure A.3: Univariate marginal and pairwise joint posteriors for dataset *nodal*. Blue, green, and red are for MCMC, SIVI with a full covariance matrix, and MFVI with a full covariance matrix.

Appendix B

Appendix for Structured Variational Inference for Community Detection

This appendix contains detailed proofs and derivation of theoretical results presented in the chaper 3, and additional experimental results. In particular, Section B.1 contains the detailed derivation of updates of the Variational Inference with Pairwise Structure (VIPS) algorithm. Section B.2 contains detailed proofs of the theoretical results presented in the chaper 3.

B.1 Detailed Derivation of the Updates of VIPS

In the chaper 3, the Evidence Lower BOund (ELBO) (3.5) for pairwise structured variational inference is

$$\begin{aligned}\mathcal{L}(Q;\pi,B) =&\frac{1}{2}\mathbb{E}_Q \sum_{i\neq j,a,b} [Z_{ia}Z_{jb}(A^{zz}_{ij}\alpha_{ab} + f(\alpha_{ab})) + Y_{ia}Y_{jb}(A^{yy}_{ij}\alpha_{ab} + f(\alpha_{ab}))] \\ &+\mathbb{E}_Q[\sum_{i\neq j,a,b} Z_{ia}Y_{jb}(A^{zy}_{ij}\alpha_{ab} + f(\alpha_{ab})) + \sum_{i,a,b} Z_{ia}Y_{ib}(A^{zy}_{ii}\alpha_{ab} + f(\alpha_{ab}))] \\ &-\sum_{i=1}^{m} \mathcal{D}_{\mathrm{KL}}(Q(z_i,y_i)||P(z_i)P(y_i))\end{aligned}$$

where $\alpha_{ab} = \log(B_{ab}/(1-B_{ab}))$ and $f(\alpha) = -\log(1+e^{\alpha})$. Denote the first four terms in ELBO as T_1, T_2, T_3, T_4, where T_1, T_2 correspond to the likelihood of the blocks A^{zz} and A^{yy} in the adjacency matrix, T_3 corresponds to the

likelihood of $(z_i, y_j), i \neq j$ and T_4 corresponds to (z_i, y_i). Plugging in the marginal density of the independent nodes in T_1, T_2, T_3 and joint density of the dependent nodes in T_4, we have

$$\begin{aligned}
T_1 =& \frac{1}{2}\sum_{i\neq j}\Big\{[(1-\phi_i)(1-\phi_j)+\phi_i\phi_j](A_{ij}^{zz}\log\frac{p}{1-p}+\log(1-p))+ \\
& [(1-\phi_i)\phi_j+\phi_i(1-\phi_j)](A_{ij}^{zz}\log\frac{q}{1-q}+\log(1-q))\Big\} \qquad \text{(B.1)} \\
T_2 =& \frac{1}{2}\sum_{i\neq j}\Big\{[(1-\xi_i)(1-\xi_j)+\xi_i\xi_j](A_{ij}^{yy}\log\frac{p}{1-p}+\log(1-p))+ \\
& [(1-\xi_i)\xi_j+\xi_i(1-\xi_j)](A_{ij}^{yy}\log\frac{q}{1-q}+\log(1-q))\Big\} \qquad \text{(B.2)} \\
T_3 =& \sum_{i\neq j}\Big\{[(1-\phi_i)(1-\xi_j)+\phi_i\xi_j](A_{ij}^{zy}\log\frac{p}{1-p}+\log(1-p))+ \\
& [(1-\phi_i)\xi_j+\phi_i(1-\xi_j)](A_{ij}^{zy}\log\frac{q}{1-q}+\log(1-q))\Big\} \qquad \text{(B.3)} \\
T_4 =& \sum_{i}\Big\{(1-\psi_i^{01}-\psi_i^{10})(A_{ii}^{zy}\log\frac{p}{1-p}+\log(1-p))+ \\
& (\psi_i^{01}+\psi_i^{10})(A_{ii}^{zy}\log\frac{q}{1-q}+\log(1-q))\Big\} \qquad \text{(B.4)}
\end{aligned}$$

The KL regularization term (3.6) is

$$\mathcal{D}_{\mathrm{KL}}(Q(z_i,y_i)||P(z_i)P(y_i)) = \sum_{0\leq c,d\leq 1}\psi_i^{cd}\log\frac{\psi_i^{cd}}{\pi^c\pi^d(1-\pi)^{1-c}(1-\pi)^{1-d}}$$

To take the derivative of $\mathcal{L}(Q;\pi,B)$ with respect to $\psi_i^{cd}, cd\neq 0$, we first have the derivative of the KL term

$$\begin{aligned}
\frac{\partial}{\partial\psi_i^{cd}}\mathcal{D}_{\mathrm{KL}}(Q(z_i,y_i)||P(z_i)P(y_i)) &= \log\frac{\psi_i^{cd}}{\pi^{c+d}(1-\pi)^{2-c-d}} - \log\frac{\psi_i^{00}}{(1-\pi)^2} \\
&= \log\frac{\psi_i^{cd}}{1-\psi_i^{01}-\psi_i^{10}-\psi_i^{11}} \quad (\pi=\frac{1}{2}) \qquad \text{(B.5)}
\end{aligned}$$

Denote the right hand side of Eq. (B.5) as $\theta_i^{cd} \coloneqq \log \frac{\psi_i^{cd}}{1-\psi_i^{01}-\psi_i^{10}-\psi_i^{11}}$. For the reconstruction terms, denoting $T(a,p) \coloneqq a\log(\frac{p}{1-p}) + \log(1-p)$ for simplicity, the derivative can be computed as

$$\frac{\partial}{\partial \psi_i^{10}}(\sum T_k) = \sum_{j,j\neq i}\Big[(2\phi_j - 1)T(A_{ij}^{zz}, p) - (2\phi_j - 1)T(A_{ij}^{zz}, q)\Big] + \tag{B.6}$$
$$\sum_{j,j\neq i}\Big[(2\xi_j - 1)T(A_{ij}^{zy}, p) - (2\xi_j - 1)T(A_{ij}^{zy}, q)\Big] + \Big[-T(A_{ii}^{zy}, p) + T(A_{ii}^{zy}, q)\Big]$$

$$\frac{\partial}{\partial \psi_i^{01}}(\sum T_k) = \sum_{j,j\neq i}\Big[(2\xi_j - 1)T(A_{ij}^{yy}, p) - (2\xi_j - 1)T(A_{ij}^{yy}, q)\Big] + \tag{B.7}$$
$$\sum_{j,j\neq i}\Big[(2\phi_j - 1)T(A_{ji}^{zy}, p) - (2\phi_j - 1)T(A_{ji}^{zy}, q)\Big] + \Big[-T(A_{ii}^{zy}, p) + T(A_{ii}^{zy}, q)\Big]$$

$$\begin{aligned}\frac{\partial}{\partial \psi_i^{11}}(\sum T_k) = &\sum_{j,j\neq i}\Big[(2\phi_j - 1)T(A_{ij}^{zz}, p) - (2\phi_j - 1)T(A_{ij}^{zz}, q)\Big] + \\ &\sum_{j,j\neq i}\Big[(2\xi_j - 1)T(A_{ij}^{yy}, p) - (2\xi_j - 1)T(A_{ij}^{yy}, q)\Big] + \\ &\sum_{j,j\neq i}\Big[(2\xi_j - 1)T(A_{ij}^{zy}, p) - (2\xi_j - 1)T(A_{ij}^{zy}, q)\Big] + \\ &\sum_{j,j\neq i}\Big[(2\phi_j - 1)T(A_{ji}^{zy}, p) - (2\phi_j - 1)T(A_{ji}^{zy}, q)\Big]\end{aligned} \tag{B.8}$$

Setting the derivatives to 0 we get the update for θ as (3.9), (3.8), (3.10).

B.2 Proofs of Main Results

To prove Theroem 2, we first need a few lemmas. First we have the following lemma for the parameters p, q and λ.

Lemma 1. *If* $p \asymp q \asymp \rho_n, \rho_n \to 0$ *and* $p - q = \Omega(\rho_n)$, *then*

$$\lambda - q = \Omega(\rho_n) > 0, \quad \frac{p+q}{2} - \lambda = \Omega(\rho_n) > 0. \tag{B.9}$$

Proof. The proof follows from Proposition 2 in Sarkar et al. [126]. □

In the proof, we utilize the spectral property of the population matrix P and generalize it to the finite sample case by bounding the term related to the residual $R = A - P$. We use Berry-Esseen Theorem to bound the residual terms conditioning on u.

Lemma 2 (Berry-Esseen bound)**.** *Define*

$$r_i = \sum_{j=1}^{n} (A_{ij} - P_{ij})(u(j) - \frac{1}{2}),$$

where u *and* A *are independent.*

$$\sup_{x \in \mathbb{R}} |P\left(r_i/\sigma_u \leq x \mid u\right) - \Phi(x)| \leq \frac{C\rho_u}{\sigma_u^3},$$

where C *is a general constant,* $\Phi(\cdot)$ *is the CDF of standard Gaussian,* ρ_u *and* σ_u *depend on* u.

Proof. Since r_i is the sum of independent, mean zero random variables, the sum of the conditional variances is

$$\sigma_u^2 = \mathrm{Var}(r_i|u) = p(1-p) \sum_{i \in G_1} (u(i) - \frac{1}{2})^2 + q(1-q) \sum_{i \in G_2} (u(i) - \frac{1}{2})^2,$$

and the sum of the conditional absolute third central moments is

$$\rho_u = p(1-p)(1-2p+2p^2) \textstyle\sum_{i \in G_1} |u(i) - \frac{1}{2}|^3 + q(1-q)(1-2q+2q^2) \sum_{i \in G_2} |u(i) - \frac{1}{2}|^3.$$

The desired bound follows from the Berry-Esseen Theorem. □

The next lemma shows despite the fact that A introduces some dependency among r_i due to its symmetry, we can still treat r_i as almost iid.

Lemma 3 (McDiarmid's Inequality)**.** *Let r_i be the noise defined in Lemma 2 and let $h(r_i)$ be a bounded function with $\|h\|_\infty \leq M$. Then*

$$P\left(\left|\frac{2}{n}\sum_{i\in\mathcal{A}} h(r_i) - \mathbb{E}(h(r_i)|u)\right| > w \mid u\right) \leq \exp\left(-\frac{c_0 w^2}{nM}\right)$$

for some general constant c_0, provided $|\mathcal{A}| = \Theta_P(n)$.

Proof. The proof follows from Lemma 20 in Sarkar et al. [126]. □

Lemma 4. *Let r_i be defined as in Lemma 2 and assume A and u are independent, we have $\sup_{i\in\mathcal{A}} |r_i| = O_P(\sqrt{n\rho_n \log n})$ if the index set $|\mathcal{A}| = \Theta_P(n)$.*

Proof. Since r_i is the sum of independent bounded random variables, for all i, $r_i = O_P(\sqrt{n\rho_n})$. By Hoeffding inequility, we know for all $t > 0$

$$P(|r_i| > t) \leq \exp(-\frac{t^2}{2n\rho_n})$$

and by the union bound

$$P(\sup_i |r_i| > t) \leq \exp(C\log n - \frac{t^2}{2n\rho_n})$$

For $\forall\epsilon > 0$, let $t = C_\epsilon\sqrt{n\rho_n \log n}$ with $n^{\frac{C_\epsilon^2}{2}-1} > 1/\epsilon$, then by definition $\sup_i |r_i| = O_P(\sqrt{n\rho_n \log n})$ □

Next we have a lemma ensuring the signal in the first iteration is not too small.

Lemma 5 (Littlewood-Offord)**.** *Let* $s_1 = (p-\lambda)\sum_{i\in G_1}(u^{(0)}(i) - 1/2) + (q - \lambda)\sum_{i\in G_2}(u^{(0)}(i)-1/2)$, $s_2 = (q-\lambda)\sum_{i\in G_1}(u^{(0)}(i)-1/2)+(p-\lambda)\sum_{i\in G_2}(u^{(0)}(i)-1/2)$. *Then*

$$P\left(|s_1| \le c\right) \le B \cdot \frac{c}{\rho_n\sqrt{n}}$$

for $c > 0$ *and* B *as constant. The same bound holds for* $|s_2|, |s_1 - s_2|$.

Proof. Noting that $2u^{(0)}(i) - 1 \in \{-1, 1\}$ each with probability $1/2$, and Lemma 1, this is a direct consequence of the Littlewood-Offord bound in Erdös [29]. □

Finally, we have the following upper and lower bound for some general update ϕ_i.

Lemma 6. *Assume* ϕ_i *has the update form* $\phi_i = (a + e^{4t(s+r_i)})/(b + e^{4t(s+r_i)})$ *for* $i \in [m]$, $b > a > 0$ *and* $b - a$, $(b-a)/b$ *are of constant order.* r_i *is defined as in Lemma 2. Let set* $\mathcal{A} \subset [m]$, *with* $\Delta > 0$, *we have*

$$\sum_{i\in\mathcal{A}}\phi_i \ge |\mathcal{A}| - \frac{b-a}{b}|\mathcal{A}|\Phi(\frac{-s+\Delta}{\sigma_u}) - C'|\mathcal{A}|\frac{\rho_u}{\sigma_u^3} - C''|\mathcal{A}|e^{-4t\Delta} - O_P(\sqrt{|\mathcal{A}|}),$$

$$\sum_{i\in\mathcal{A}}\phi_i \le |\mathcal{A}| - \frac{b-a}{b}|\mathcal{A}|\Phi(\frac{-s-\Delta}{\sigma_u}) + C'|\mathcal{A}|\frac{\rho_u}{\sigma_u^3} + |\mathcal{A}|e^{-4t\Delta} + O_P(\sqrt{|\mathcal{A}|}).$$

Proof. Define the set $J^+ = \{i : r_i > -s + \Delta\}$, $\Delta \geq 0$. For $i \in \mathcal{A} \cap J^+$

$$\phi_i = \frac{a + e^{4t(s+r_i)}}{b + e^{4t(s+r_i)}} \geq \frac{a + e^{4t\Delta}}{b + e^{4t\Delta}} \geq 1 - (b-a)e^{-4t\Delta}$$

For $i \in (\mathcal{A} \cap J^+)^c$, $\phi_i \geq a/b$, therefore

$$\begin{aligned}\sum_{i\in\mathcal{A}} \phi_i \geq & |\mathcal{A} \cap J^+|(1 - (b-a)e^{-4t\Delta}) + \frac{a}{b}(|\mathcal{A}| - |\mathcal{A} \cap J^+|) \\ = & |\mathcal{A} \cap J^+|(\frac{b-a}{b} - (b-a)e^{-4t\Delta}) + \frac{a}{b}|\mathcal{A}|\end{aligned}$$

By Lemmas 2 and 3, we have

$$\begin{aligned}|\mathcal{A} \cap J^+| = & \sum_{i\in\mathcal{A}} \mathbf{1}[r_i > -s + \Delta] \\ = & |\mathcal{A}| \cdot P(r_i > -s + \Delta) + O_P(\sqrt{|\mathcal{A}|}) \\ \geq & |\mathcal{A}| \cdot (1 - \Phi(\frac{-s+\Delta}{\sigma_u}) - C_0\frac{\rho_u}{\sigma_u^3}) + O_P(\sqrt{|\mathcal{A}|}).\end{aligned}$$

Combining the above,

$$\sum_{i\in\mathcal{A}} \phi_i \geq |\mathcal{A}| - \frac{b-a}{b}|\mathcal{A}|\Phi(\frac{-s+\Delta}{\sigma_u}) - C'|\mathcal{A}|\frac{\rho_u}{\sigma_u^3} - C''|\mathcal{A}|e^{-4t\Delta} - O_P(\sqrt{|\mathcal{A}|})$$

Similarly, define the set $J^- = \{i : r_i < -s - \Delta\}$, $\Delta \geq 0$. For $i \in \mathcal{A} \cap J^-$,

$$\phi_i = \frac{a + e^{4t(s+r_i)}}{b + e^{4t(s+r_i)}} \leq \frac{a + e^{-4t\Delta}}{b + e^{-4t\Delta}} \leq \frac{a}{b} + e^{-4t\Delta}$$

For $i \in (\mathcal{A} \cap J^-)^c$, $\phi_i \leq 1$, so

$$\begin{aligned}\sum_{i\in\mathcal{A}} \phi_i \leq & |\mathcal{A} \cap J^-|(\frac{a}{b} + e^{-4t\Delta}) + (|\mathcal{A}| - |\mathcal{A} \cap J^-|) \\ = & |\mathcal{A}| - |\mathcal{A} \cap J^-|(1 - \frac{a}{b} - e^{-4t\Delta}) + O_P(\sqrt{|\mathcal{A}|})\end{aligned}$$

By Lemmas 2 and 3,

$$|\mathcal{A} \cap J^-| \geq |\mathcal{A}| \cdot (\Phi(\frac{-s-\Delta}{\sigma_u}) - C_0\frac{\rho_u}{\sigma_u^3}) - O_P(\sqrt{|\mathcal{A}|})$$

so

$$\sum_{i\in\mathcal{A}} \phi_i \leq |\mathcal{A}| - \frac{b-a}{b}|\mathcal{A}|\Phi(\frac{-s-\Delta}{\sigma_u}) + C'|\mathcal{A}|\frac{\rho_u}{\sigma_u^3} + |\mathcal{A}|e^{-4t\Delta} + O_P(\sqrt{|\mathcal{A}|})$$

□

Proof of Theorem 2. Throughout the proof, we assume A has self-loops for convenience, which does not affect the asymptotic results.

Analysis of the first iteration in the first meta iteration:

For random initialized $u^{(0)}$, the initial signal $|\langle u^{(0)}, v_2\rangle| = O_P(\sqrt{n})$. Using the graph split $A^{(0)}$, we write the update of θ^{10} as

$$\begin{aligned}\theta^{10} =& 4t([6(A^{(0)})^{zz}, 6(A^{(0)})^{zy}] - \lambda J)(u^{(0)} - \frac{1}{2}\mathbf{1}_n) \\ =& \underbrace{4t([P^{zz}, P^{zy}] - \lambda J)(u^{(0)} - \frac{1}{2}\mathbf{1}_n)}_{\text{signal}} + \underbrace{4t[6(A^{(0)})^{zz} - P^{zz}, 6(A^{(0)})^{zy} - P^{zy}](u^{(0)} - \frac{1}{2}\mathbf{1}_n)}_{\text{noise}},\end{aligned} \tag{B.10}$$

where P is the population matrix of A. Denote $R^{(0)} = 6A^{(0)} - P$ and $r^{(0)} = [(R^{(0)})^{zz}, (R^{(0)})^{zy}](u^{(0)} - \frac{1}{2}\mathbf{1})$ Since P has singular value decomposition as $P = \frac{p+q}{2}\mathbf{1}_n\mathbf{1}_n^T + \frac{p-q}{2}v_2v_2^T$, the signal part is blockwise constant and we can write

$$\theta^{10} = 4t(s_1\mathbf{1}_{C_1} + s_2\mathbf{1}_{C_2} + r^{(0)}), \tag{B.11}$$

where

$$\begin{aligned}s_1 =& (\frac{p+q}{2} - \lambda)(\langle u^{(0)}, \mathbf{1}_n\rangle - m) + (\frac{p-q}{2})\langle u^{(0)}, v_2\rangle \\ s_2 =& (\frac{p+q}{2} - \lambda)(\langle u^{(0)}, \mathbf{1}_n\rangle - m) - (\frac{p-q}{2})\langle u^{(0)}, v_2\rangle\end{aligned} \tag{B.12}$$

By (3.11), since we initialize with $\theta^{01}, \theta^{11} = 0$, the marginal probabilities are updated as

$$\phi_1^{(1)} = \frac{1 + e^{\theta^{10}}}{3 + e^{\theta^{10}}}, \quad \xi_1^{(1)} = \frac{2}{3 + e^{\theta^{10}}} \tag{B.13}$$

Next we show the signal $|\langle u, v_2 \rangle|$ increases from $O_P(\sqrt{n})$ to $\Omega_P(n\sqrt{\rho_n})$. (We omit the superscript on logits s,x and y now for simplicity.) Since

$$\langle u_1^{(1)}, v_2 \rangle = \langle \phi_{1i}^{(1)}, v_{21} \rangle + \langle \xi_{1i}^{(1)}, v_{22} \rangle = \sum_{i \in C_1} \phi_i^{(1)} - \sum_{i \in C_2} \phi_i^{(1)} + \langle \xi^{(1)}, v_{22} \rangle$$

we use Lemma 6 to bound $\sum_{i \in C_1} \phi_i^{(1)}$ and $\sum_{i \in C_2} \phi_i^{(1)}$. Since s_1 and s_2 depends on $u^{(0)}$, we consider two cases conditioning on $u^{(0)}$.

Case 1: $s_1 > s_2$. By Lemma 6, let $\Delta = \frac{1}{4}(s_1 - s_2)$ with $\mathcal{A} = C_1, C_2$, $(a, b) = (1, 3)$, conditioning on $u^{(0)}$,

$$\sum_{i \in C_1} \phi_{1i}^{(1)} \geq -\frac{n}{6}\Phi(-\frac{s_1 - \frac{1}{4}(s_1 - s_2)}{\sigma_u}) + \frac{n}{4} - C'n\frac{\rho_u}{\sigma_u^3} - C''ne^{-t(s_1 - s_2)} - O_P(\sqrt{n}),$$

$$\sum_{i \in C_2} \phi_{1i}^{(1)} \leq \frac{n}{4} - \frac{n}{6}\Phi(-\frac{s_2 + \frac{1}{4}(s_1 - s_2)}{\sigma_u}) + C'n\frac{\rho_u}{\sigma_u^3} + C''ne^{-t(s_1 - s_2)} + O_P(\sqrt{n}),$$

where the $O_P(\sqrt{n})$ term can be made uniform in $u^{(0)}$. So we have

$$\begin{aligned}
\langle \phi_1^{(1)}, v_{21} \rangle \geq & \frac{n}{6}(\Phi(-\frac{s_2 + \frac{1}{4}(s_1 - s_2)}{\sigma_u}) - \Phi(-\frac{s_1 - \frac{1}{4}(s_1 - s_2)}{\sigma_u})) \\
& - C'n\frac{\rho_u}{\sigma_u^3} - C''ne^{-t(s_1 - s_2)} - O_P(\sqrt{n}) \\
\geq & \frac{n}{6\sqrt{2\pi}}(\frac{s_1 - s_2}{2\sigma_u}) \exp\big(-\frac{s_1^2 \vee s_2^2}{2\sigma_u^2}\big) \\
& - C'n\frac{\rho_u}{\sigma_u^3} - C''ne^{-t(s_1 - s_2)} - O_P(\sqrt{n}).
\end{aligned} \tag{B.14}$$

Here to approximate the CDF Φ, we have used

$$|\Phi(x) - 1/2| = \frac{1}{\sqrt{2\pi}} \int_0^{|x|} e^{-u^2/2} du \geq \frac{|x|}{\sqrt{2\pi}} e^{-x^2/2}. \tag{B.15}$$

Case 2: $s_1 < s_2$. The same analysis applies with s_1 and s_2 interchanged. Combining *Case 1* and *Case 2*, for any given $u^{(0)}$,

$$\begin{aligned} |\langle \phi_1^{(1)}, v_{21} \rangle| \geq & \frac{n}{6\sqrt{2\pi}} \left(\frac{|s_1 - s_2|}{2\sigma_u}\right) \exp\left(-\frac{s_1^2 \vee s_2^2}{2\sigma_u^2}\right) \\ & - C' n \frac{\rho_u}{\sigma_u^3} - C'' n e^{-t|s_1 - s_2|} - O_P(\sqrt{n}). \end{aligned} \tag{B.16}$$

We note that $|s_1|$, $|s_2|$, $|s_1 - s_2|$ are of order $\Omega_P(\sqrt{n}\rho_n)$ by Lemma 5. Also $\sigma_u^2, \rho_u \asymp n\rho_n$, $e^{-4t|s_1 - s_2|} = \exp(-\Omega(\rho_n \sqrt{n}))$. We can conclude that $|\langle \phi_1^{(1)}, v_{21} \rangle|$ $= \Omega_P(n\sqrt{\rho_n})$. For $\langle \xi_1^{(1)}, v_{22} \rangle$ we have

$$\begin{aligned} |\langle \xi_1^{(1)}, v_{22} \rangle| = & \Big| \sum_{i \in C_1'} \xi_i^{(1)} - \sum_{i \in C_2'} \xi_i^{(1)} \Big| = \Big| \sum_{i \in C_2'} \phi_i^{(1)} - \sum_{i \in C_1'} \phi_i^{(1)} + |C_1'| - |C_2'| \Big| \\ = & O_P(\sqrt{n}) \end{aligned}$$

Therefore we have $|\langle u_1^{(1)}, v_2 \rangle| = \Omega_P(n\sqrt{\rho_n})$. By (B.13), $\langle u_1^{(1)}, \mathbf{1} \rangle - m = 0$.

Due to the symmetry in s_1 and s_2, WLOG in the following analysis, we assume $\langle u_1^{(1)}, v_2 \rangle > 0$ (equivalently $s_1 > s_2$).

Analysis of the second iteration in the first meta iteration:

Similar to (B.10), we can write

$$\begin{aligned}\theta^{01} =& 4t([6(A^{(1)})^{yz}, 6(A^{(1)})^{yy}] - \lambda J)(u_1^{(1)} - \frac{1}{2}\mathbf{1}_n) \\ =& \underbrace{4t([P^{yz}, P^{yy}] - \lambda J)(u_1^{(1)} - \frac{1}{2}\mathbf{1}_n)}_{\text{signal}} \\ &+ \underbrace{4t(R^{(1)})^{yz}(\phi_1^{(1)} - \frac{1}{2}\mathbf{1}_m) + 4t(R^{(1)})^{yy}(\xi_1^{(1)} - \frac{1}{2}\mathbf{1}_m)}_{\text{noise} \coloneqq 4tr_i^{(1)}}.\end{aligned}$$

Noting the signal part is blockwise constant, we have

$$\begin{aligned}\theta^{01} =& 4t(x_1\mathbf{1}_{C_1'} + x_2\mathbf{1}_{C_2'} + r^{(1)}), \\ x_1 =& (\frac{p+q}{2} - \lambda)(\langle u_1^{(1)}, \mathbf{1}_n\rangle - m) + (\frac{p-q}{2})\langle u_1^{(1)}, v_2\rangle \\ x_2 =& (\frac{p+q}{2} - \lambda)(\langle u_1^{(1)}, \mathbf{1}_n\rangle - m) - (\frac{p-q}{2})\langle u_1^{(1)}, v_2\rangle\end{aligned}$$

By (B.13), $\langle u_1^{(1)}, \mathbf{1}_n\rangle - m = 0$ and we have

$$x_1 = (\frac{p-q}{2})\langle u_1^{(1)}, v_2\rangle, \quad x_2 = -x_1.$$

It follows then from the first iteration that $x_1, -x_2 = \Omega_P(n\rho_n^{3/2})$. The update for $u_2^{(1)}$ is

$$\phi_2^{(1)} = \frac{1 + e^{\theta^{10}}}{2 + e^{\theta^{10}} + e^{\theta^{01}}}, \quad \xi_2^{(1)} = \frac{1 + e^{\theta^{01}}}{2 + e^{\theta^{10}} + e^{\theta^{01}}} \tag{B.17}$$

Since the signal part of θ^{10} and θ^{01} are blockwise constant on C_1, C_2 and C_1',

C_2' respectively, $\langle u_2^{(1)}, v_2\rangle$ can be calculated as

$$\langle \phi_2^{(1)}, v_{21}\rangle = \sum_{i\in C_{11}} \frac{1+e^{4t(s_1+r_i^{(0)})}}{2+e^{4t(s_1+r_i^{(0)})}+e^{4t(x_1+r_i^{(1)})}} + \sum_{i\in C_{12}} \frac{1+e^{4t(s_1+r_i^{(0)})}}{2+e^{4t(s_1+r_i^{(0)})}+e^{4t(x_2+r_i^{(1)})}}$$
$$- \sum_{i\in C_{21}} \frac{1+e^{4t(s_2+r_i^{(0)})}}{2+e^{4t(s_2+r_i^{(0)})}+e^{4t(x_1+r_i^{(1)})}} - \sum_{i\in C_{22}} \frac{1+e^{4t(s_2+r_i^{(0)})}}{2+e^{4t(s_2+r_i^{(0)})}+e^{4t(x_2+r_i^{(1)})}}$$

$$\langle \xi_2^{(1)}, v_{22}\rangle = \sum_{i\in C_{11}} \frac{1+e^{4t(x_1+r_i^{(1)})}}{2+e^{4t(s_1+r_i^{(0)})}+e^{4t(x_1+r_i^{(1)})}} + \sum_{i\in C_{21}} \frac{1+e^{4t(x_1+r_i^{(1)})}}{2+e^{4t(s_2+r_i^{(0)})}+e^{4t(x_1+r_i^{(1)})}}$$
$$- \sum_{i\in C_{12}} \frac{1+e^{4t(x_2+r_i^{(1)})}}{2+e^{4t(s_1+r_i^{(0)})}+e^{4t(x_2+r_i^{(1)})}} - \sum_{i\in C_{22}} \frac{1+e^{4t(x_2+r_i^{(1)})}}{2+e^{4t(s_2+r_i^{(0)})}+e^{4t(x_2+r_i^{(1)})}}$$

In the case of $\langle u_1^{(1)}, v_2\rangle > 0$, we know $s_1 > s_2$ and $x_1 > 0 > x_2$. We first show that $\langle \phi_2^{(1)}, v_{21}\rangle$ is positive by finding a lower bound for the summations over C_{12}, C_{21}, C_{22} (since the sum over C_{11} is always positive).

For the summation over C_{12}, note that $|x_2|$ dominates both s_1 and $r_i^{(0)}$, $r_i^{(1)}$ by Lemma 4, we have

$$\sum_{i\in C_{12}} \frac{1+e^{4t(s_1+r_i^{(0)})}}{2+e^{4t(s_1+r_i^{(0)})}+e^{4t(x_2+r_i^{(1)})}} = \sum_{i\in C_{12}} \frac{1+e^{4t(s_1+r_i^{(0)})}}{2+e^{4t(s_1+r_i^{(0)})}} + n\exp(-\Omega_P(n\rho_n^{3/2})).$$

To lower bound the first term, we use Lemma 6 by first conditioning on $u^{(0)}$,

$$\textstyle\sum_{i\in C_{12}} \frac{1+e^{4t(s_1+r_i^{(0)})}}{2+e^{4t(s_1+r_i^{(0)})}} \geq \frac{n}{8}\left(1-\frac{1}{2}\Phi(\frac{-s_1+\Delta}{\sigma_u})\right) - C'n\frac{\rho_u}{\sigma_u^3} - C''ne^{-4t\Delta} - O_P(\sqrt{n}) \tag{B.18}$$

For the summation over C_{22},

$$\sum_{i\in C_{22}} \frac{1+e^{4t(s_2+r_i^{(0)})}}{2+e^{4t(s_2+r_i^{(0)})}+e^{4t(x_2+r_i^{(1)})}} \leq \sum_{i\in C_{22}} \frac{1+e^{4t(s_2+r_i^{(0)})}}{2+e^{4t(s_2+r_i^{(0)})}}$$
$$\leq \frac{n}{8}\left(1-\frac{1}{2}\Phi(\frac{-s_2-\Delta}{\sigma_u})\right) + C'n\frac{\rho_u}{\sigma_u^3} + C''ne^{-4t\Delta} + O_P(\sqrt{n}) \tag{B.19}$$

For the summation over C_{21}, x_1 dominates s_2 and $r_i^{(0)}, r_i^{(1)}$ by Lemma 4,

$$\sum_{i\in C_{21}} \frac{1+e^{4t(s_2+r_i^{(0)})}}{2+e^{4t(s_2+r_i^{(0)})}+e^{4t(x_1+r_i^{(1)})}} = n\exp(-\Omega_P(n\rho_n^{3/2})). \tag{B.20}$$

Combining (B.18) - (B.20), setting $\Delta = \frac{1}{4}(s_1 - s_2)$, we have

$$\begin{aligned}
&\langle \phi_2^{(1)}, v_{21}\rangle \\
&\geq \frac{n}{8}\big[\frac{1}{2}\Phi(\frac{-s_2-\Delta}{\sigma_u}) - \frac{1}{2}\Phi(\frac{-s_1+\Delta}{\sigma_u})\big] - C'n\frac{\rho_u}{\sigma_u^3} - C''ne^{-4t\Delta} - O_P(\sqrt{n}) \\
\geq &\frac{n}{16}\frac{1}{\sqrt{2\pi}}(\frac{s_1-s_2}{\sigma_u})\exp\big(-\frac{s_1^2\vee s_2^2}{2\sigma_u^2}\big) - C'n\frac{\rho_u}{\sigma_u^3} - C''ne^{-t(s_1-s_2)} - O_P(\sqrt{n})
\end{aligned}$$

by the same argument as (B.14). As before, we can see that

$$\langle \phi_2^{(1)}, v_{21}\rangle = \Omega_P(n\sqrt{\rho_n})$$

For $\langle \xi_2^{(1)}, v_{22}\rangle$, since $(1+e^x)/(2+e^x) \leq 1/2 + e^x$, we have

$$\begin{aligned}
&\sum_{i\in C_{12}} \frac{1+e^{4t(x_2+r_i^{(1)})}}{2+e^{4t(s_1+r_i^{(0)})}+e^{4t(x_2+r_i^{(1)})}} + \sum_{i\in C_{22}} \frac{1+e^{4t(x_2+r_i^{(1)})}}{2+e^{4t(s_2+r_i^{(0)})}+e^{4t(x_2+r_i^{(1)})}} \\
\leq &\frac{n}{8} + \sum_{i\in C_2'} e^{4t(x_2+r_i^{(1)})} + O_P(\sqrt{n}) \\
\leq &\frac{n}{8} + O_P(\sqrt{n}).
\end{aligned} \tag{B.21}$$

For the other two sums, we have

$$\begin{aligned}
\sum_{i\in C_{11}} \frac{1+e^{4t(x_1+r_i^{(1)})}}{2+e^{4t(s_1+r_i^{(0)})}+e^{4t(x_1+r_i^{(1)})}} &\geq \frac{n}{8} - O_P(\sqrt{n}) - n\exp(-\Omega_P(n\rho_n^{3/2})), \\
&\geq \frac{n}{8} - O_P(\sqrt{n})
\end{aligned} \tag{B.22}$$

and

$$\sum_{i\in C_{21}} \frac{1+e^{4t(x_1+r_i^{(1)})}}{2+e^{4t(s_2+r_i^{(0)})}+e^{4t(x_1+r_i^{(1)})}} \geq \frac{n}{8} - O_P(\sqrt{n}) \tag{B.23}$$

Equations (B.21) - (B.23) imply

$$\langle \xi_2^{(1)}, v_{22} \rangle \geq \frac{n}{8} - O_P(\sqrt{n}).$$

Therefore $\langle u_2^{(1)}, v_2 \rangle \geq n/8 - O_P(\sqrt{n})$. Since by (B.17), $\phi_2^{(1)} = \mathbf{1}_m - \xi_2^{(1)}$, the inner product $\langle u_2^{(1)}, \mathbf{1} \rangle - m = 0$.

Analysis of the third iteration in the first meta iteration:

Similar to the previous two iterations, we can write

$$\theta^{11} = 4t(y_1 \mathbf{1}_{C_1} + y_2 \mathbf{1}_{C_2} + y_1 \mathbf{1}_{C_1'} + y_2 \mathbf{1}_{C_2'} + r^{(2)}),$$
$$y_1 = (\frac{p-q}{2}) \langle u_2^{(1)}, v_2 \rangle, \quad y_2 = -y_1$$
$$r^{(2)} = [(R^{(2)})^{zz}, (R^{(2)})^{zy}](u_2 - \frac{1}{2}\mathbf{1}_n) + [(R^{(2)})^{yz}, (R^{(2)})^{yy}](u_2^{(1)} - \frac{1}{2}\mathbf{1}_n).$$

It follows from the second iteration that $y_1, -y_2 = \Omega_P(n\rho_n)$. The update for $u_3^{(1)}$ is

$$\phi_3^{(1)} = \frac{e^{\theta^{11}} + e^{\theta^{10}}}{1 + e^{\theta^{10}} + e^{\theta^{01}} + e^{\theta^{11}}}, \quad \xi_3^{(1)} = \frac{e^{\theta^{11}} + e^{\theta^{01}}}{1 + e^{\theta^{10}} + e^{\theta^{01}} + e^{\theta^{11}}} \tag{B.24}$$

The $\langle u_3^{(1)}, v_2 \rangle$ can be calculated as

$$\begin{aligned}
\langle u_3^{(1)}, v_2 \rangle = & \sum_{i \in C_{11}} \frac{2e^{8t(y_1 + r_i^{(2)})} + e^{4t(x_1 + r_i^{(1)})} + e^{4t(s_1 + r_i^{(0)})}}{1 + e^{4t(s_1 + r_i^{(0)})} + e^{4t(x_1 + r_i^{(1)})} + e^{8t(y_1 + r_i^{(2)})}} \\
& + \sum_{i \in C_{12}} \frac{e^{4t(s_1 + r_i^{(0)})} - e^{4t(x_2 + r_i^{(1)})}}{1 + e^{4tr_i^{(2)}} + e^{4t(s_1 + r_i^{(0)})} + e^{4t(x_2 + r_i^{(1)})}} \\
& + \sum_{i \in C_{21}} \frac{e^{4t(x_1 + r_i^{(1)})} - e^{4t(s_2 + r_i^{(0)})}}{1 + e^{4tr_i^{(2)}} + e^{4t(s_2 + r_i^{(0)})} + e^{4t(x_1 + r_i^{(1)})}} \\
& - \sum_{i \in C_{22}} \frac{2e^{8t(y_2 + r_i^{(2)})} + e^{4t(s_2 + r_i^{(0)})} + e^{4t(x_2 + r_i^{(1)})}}{1 + e^{4t(s_2 + r_i^{(0)})} + e^{4t(x_2 + r_i^{(1)})} + e^{8t(y_2 + r_i^{(2)})}}
\end{aligned} \tag{B.25}$$

Using the order of the x and y terms and Lemma 4, we can lower bound $\langle u_3^{(1)}, v_2 \rangle$ by

$$\begin{aligned}\langle u_3^{(1)}, v_2 \rangle &\geq \tfrac{n}{4} + \textstyle\sum_{i \in C_{12}} \frac{e^{4t(s_1+r_i^{(0)})}}{1+e^{4tr_i^{(2)}}+e^{4t(s_1+r_i^{(0)})}} + \tfrac{n}{8} - \sum_{i \in C_{22}} \frac{e^{4t(s_2+r_i^{(0)})}}{1+e^{4t(s_2+r_i^{(0)})}} - O_P(\sqrt{n}) \\ &\geq \tfrac{n}{4} - O_P(\sqrt{n}). \end{aligned} \tag{B.26}$$

Next we bound $\langle u_3^{(1)}, \mathbf{1}_n \rangle - m$.

$$\begin{aligned}\langle u_3^{(1)}, \mathbf{1}_n \rangle = &\sum_{i \in C_{11}} \frac{2e^{8t(y_1+r_i^{(2)})} + e^{4t(x_1+r_i^{(1)})} + e^{4t(s_1+r_i^{(0)})}}{1 + e^{4t(s_1+r_i^{(0)})} + e^{4t(x_1+r_i^{(1)})} + e^{8t(y_1+r_i^{(2)})}} \\ &+ \sum_{i \in C_{12}} \frac{2e^{4tr_i^{(2)}} + e^{4t(s_1+r_i^{(0)})} + e^{4t(x_2+r_i^{(1)})}}{1 + e^{4tr_i^{(2)}} + e^{4t(s_1+r_i^{(0)})} + e^{4t(x_2+r_i^{(1)})}} \\ &+ \sum_{i \in C_{21}} \frac{2e^{4tr_i^{(2)}} + e^{4t(x_1+r_i^{(1)})} + e^{4t(s_2+r_i^{(0)})}}{1 + e^{4tr_i^{(2)}} + e^{4t(s_2+r_i^{(0)})} + e^{4t(x_1+r_i^{(1)})}} \\ &+ \sum_{i \in C_{22}} \frac{2e^{8t(y_2+r_i^{(2)})} + e^{4t(s_2+r_i^{(0)})} + e^{4t(x_2+r_i^{(1)})}}{1 + e^{4t(s_2+r_i^{(0)})} + e^{4t(x_2+r_i^{(1)})} + e^{8t(y_2+r_i^{(2)})}}, \end{aligned} \tag{B.27}$$

Then

$$\langle u_3^{(1)}, \mathbf{1}_n \rangle = \tfrac{3n}{8} + \textstyle\sum_{i \in C_{12}} \frac{2e^{4tr_i^{(2)}}+e^{4t(s_1+r_i^{(0)})}}{1+e^{4tr_i^{(2)}}+e^{4t(s_1+r_i^{(0)})}} + \sum_{i \in C_{22}} \frac{e^{4t(s_2+r_i^{(0)})}}{1+e^{4t(s_2+r_i^{(0)})}} + O_P(\sqrt{n}),$$

$$\langle u_3^{(1)}, \mathbf{1}_n \rangle \geq \frac{3n}{8} - O_P(\sqrt{n}),$$

and

$$\begin{aligned}\langle u_3^{(1)}, \mathbf{1}_n \rangle &\leq \frac{3n}{8} + \sum_{i \in C_{12}} \left(\frac{e^{4tr_i^{(2)}}}{1 + e^{4tr_i^{(2)}} + e^{4t(s_1+r_i^{(0)})}} + \frac{e^{4tr_i^{(2)}} + e^{4t(s_1+r_i^{(0)})}}{1 + e^{4tr_i^{(2)}} + e^{4t(s_1+r_i^{(0)})}} \right) \\ &\qquad + \sum_{i \in C_{22}} \frac{e^{4t(s_2+r_i^{(0)})}}{1 + e^{4t(s_2+r_i^{(0)})}} + O_P(\sqrt{n}) \\ &\leq \frac{3n}{4} + O_P(\sqrt{n}) \end{aligned}$$

It follows then

$$-n/8 - O_P(\sqrt{n}) \leq \langle u_3^{(1)}, \mathbf{1}_n \rangle - m \leq n/4 + O_P(\sqrt{n}). \tag{B.28}$$

Analysis of the second meta iteration:

We first show that from the previous iteration, the signal $\langle u_3, v_2 \rangle$ will always dominate $|\langle u_3, \mathbf{1}_n \rangle - m|$ which gives desired sign and magnitude of the logits. Then we show the algorithm converges to the true labels after the second meta iteration. Using the same decomposition as (B.11),

$$\begin{aligned} s_1^{(2)} &= (\frac{p+q}{2} - \lambda)(\langle u_3^{(1)}, \mathbf{1}_n \rangle - m) + \frac{p-q}{2}\langle u_3^{(1)}, v_2 \rangle \qquad \text{(B.29)} \\ &\geq -\frac{n}{8}(\frac{p+q}{2} - \lambda) + \frac{n}{4} \cdot \frac{p-q}{2} - o_P(n\rho_n) \\ &\geq \frac{n}{8}(\lambda - q) - o_P(n\rho_n) \\ s_2^{(2)} &= (\frac{p+q}{2} - \lambda)(\langle u_3^{(1)}, \mathbf{1}_n \rangle - m) - \frac{p-q}{2}\langle u_3^{(1)}, v_2 \rangle \\ &\leq \frac{n}{4}(\frac{p+q}{2} - \lambda) - \frac{n}{4} \cdot \frac{p-q}{2} + o_P(n\rho_n) \\ &= -\frac{n}{4}(\lambda - q) + o_P(n\rho_n), \qquad \text{(B.30)} \end{aligned}$$

where we have used Lemma 1. After the first meta iteration, the logits satisfy

$$s_1^{(2)}, -s_2^{(2)} = \Omega_P(n\rho_n),\ x_1^{(1)}, -x_2^{(1)} = \Omega_P(n\rho_n^{\frac{3}{2}}),\ y_1^{(1)}, -y_2^{(1)} = \Omega_P(n\rho_n).$$

Here we have added the superscripts for the first meta iteration for clarity.

In the first iteration of the second meta iteration, $\langle u_1^{(2)}, v_2 \rangle$ is computed as (B.25) with s_1 and s_2 replaced with $s_1^{(2)}$ and $s_2^{(2)}$ and the noise replaced accordingly. It is easy to see that

$$\langle u_1^{(2)}, v_2 \rangle \geq \frac{3n}{8} - o_P(n).$$

Similarly from (B.27),

$$-\frac{n}{8} - o_P(n) \leq \langle u_1^{(2)}, \mathbf{1}_n \rangle - m \leq o_P(n).$$

The logits are updated as $(\frac{p+q}{2} - \lambda)(\langle u_1^{(2)}, \mathbf{1}_n \rangle - m) \pm \frac{p-q}{2} \langle u_1^{(2)}, v_2 \rangle$, so

$$x_1^{(2)}, -x_2^{(2)} = \Omega_P(n\rho_n), \tag{B.31}$$

The same analysis and results hold for $u_2^{(2)}$ and $(y_1^{(2)}, y_2^{(2)})$. We now show after the second meta iteration, in addition to the condition (B.31), we further have

$$2y_1^{(2)} - s_1^{(2)} = \Omega_P(n\rho_n), \quad 2y_1^{(2)} - x_1^{(2)} = \Omega_P(n\rho_n) \tag{B.32}$$

To simplify notation, let

$$\alpha_i(s_1, x_1, y_1) \coloneqq \frac{2e^{8t(y_1 + r_i^{(y)})} + e^{4t(x_1 + r_i^{(x)})} + e^{4t(s_1 + r_i^{(s)})}}{1 + e^{4t(s_1 + r_i^{(s)})} + e^{4t(x_1 + r_i^{(x)})} + e^{8t(y_1 + r_i^{(y)})}}$$

where r's are the noise associated with each signal and we have Lemma 4 bounding their order uniformly.

We first provide an upper bound on $\langle u_3^{(1)}, v_2 \rangle$. In (B.25), by Lemma 6

$$\begin{aligned}
\langle u_3^{(1)}, v_2 \rangle \leq & \frac{n}{4} + \sum_{i \in C_{12}} \frac{e^{4t(s_1^{(1)} + r_i^{(0)})}}{1 + e^{4t(s_1^{(1)} + r_i^{(0)})}} + \frac{n}{8} - \sum_{i \in C_{22}} \frac{e^{4t(s_2^{(1)} + r_i^{(0)})}}{1 + e^{4t(s_2^{(1)} + r_i^{(0)})}} + O_P(\sqrt{n}) \\
\leq & \tfrac{3n}{8} + \tfrac{n}{8} \left(\Phi(\tfrac{-s_2^{(1)} + \Delta}{\sigma_u}) - \Phi(\tfrac{-s_1^{(1)} - \Delta}{\sigma_u}) \right) + C' n \tfrac{\rho_u}{\sigma_u^3} + C'' n e^{-4t\Delta} + O_P(\sqrt{n}) \\
\leq & \frac{3n}{8} + o_P(n).
\end{aligned} \tag{B.33}$$

For $u_1^{(2)}$, based on (B.25) and (B.27),

$$\begin{aligned}
\langle u_1^{(2)}, v_2 \rangle &= \sum_{i \in C_{11}} \alpha_i(s_1^{(2)}, x_1^{(1)}, y_1^{(1)}) + \frac{n}{4} - o_P(n), \\
\langle u_1^{(2)}, \mathbf{1}_n \rangle - m &= \sum_{i \in C_{11}} \alpha_i(s_1^{(2)}, x_1^{(1)}, y_1^{(1)}) - \frac{n}{4} - o_P(n).
\end{aligned}$$

$$\text{Similarly,}\quad \langle u_2^{(2)}, v_2\rangle = \sum_{i\in C_{11}} \alpha_i(s_1^{(2)}, x_1^{(2)}, y_1^{(1)}) + \frac{n}{4} - o_P(n),$$
$$\langle u_2^{(2)}, \mathbf{1}_n\rangle - m = \sum_{i\in C_{11}} \alpha_i(s_1^{(2)}, x_1^{(2)}, y_1^{(1)}) - \frac{n}{4} - o_P(n).$$

For convenience denote $a = \frac{p+q}{2} - \lambda$ and $b = \frac{p-q}{2}$, then we have

$$\begin{aligned}
2y_1^{(2)} - s_1^{(2)} &= a(2\langle u_2^{(2)}, \mathbf{1}_n\rangle - \langle u_3^{(1)}, \mathbf{1}_n\rangle - m) + b(2\langle u_2^{(2)}, v_2\rangle - \langle u_3^{(1)}, v_2\rangle) \\
&\geq a\left(2\sum_{i\in C_{11}} \alpha_i(s_1^{(2)}, x_1^{(2)}, y_1^{(1)}) - \frac{n}{4} - m\right) \\
&\quad + b\left(2\sum_{i\in C_{11}} \alpha_i(s_1^{(2)}, x_1^{(2)}, y_1^{(1)}) + \frac{n}{2} - \frac{3n}{8}\right) - o_P(n\rho_n) \\
&= 2(a+b)\sum_{i\in C_{11}} \alpha_i(s_1^{(2)}, x_1^{(2)}, y_1^{(1)}) - \frac{3an}{8} + \frac{bn}{8} - o_P(n\rho_n)
\end{aligned}$$

by (B.33) and (B.28). Since $\alpha_i(s_1^{(2)}, x_1^{(2)}, y_1^{(1)}) \geq 1 + o_P(1)$, we can conclude

$$2y_1^{(2)} - s_1^{(2)} \geq \frac{3bn}{8} - \frac{an}{8} - o_P(n\rho_n) = \Omega(n\rho_n).$$

Similarly, we can check that

$$\begin{aligned}
&2y_1^{(2)} - x_1^{(2)} = a(2\langle u_2^{(2)}, \mathbf{1}_n\rangle - \langle u_1^{(2)}, \mathbf{1}_n\rangle - m) + b(2\langle u_2^{(2)}, v_2\rangle - \langle u_1^{(2)}, v_2\rangle) \\
=&(a+b)\sum_{i\in C_{11}} [2\alpha_i(s_1^{(2)}, x_1^{(2)}, y_1^{(1)}) - \alpha_i(s_1^{(2)}, x_1^{(1)}, y_1^{(1)})] - \frac{(a-b)n}{4} + o_P(n\rho_n) \\
\geq&\frac{(b-a)n}{4} - o_P(n\rho_n) = \Omega(n\rho_n)
\end{aligned} \tag{B.34}$$

as $\alpha_i(s_1^{(2)}, x_1^{(2)}, y_1^{(1)}) > \alpha_i(s_1^{(2)}, x_1^{(1)}, y_1^{(1)})$. Thus condition (B.32) holds.

Now we need to analyze the third iteration in this meta iteration. Since $\alpha_i(s_1^{(2)}, x_1^{(2)}, y_1^{(1)}) \leq 2$, with (B.30)

$$\begin{aligned}
y_1^{(2)} + y_2^{(2)} &= 2a(\langle u_2^{(2)}, \mathbf{1}_n\rangle - m) = o_P(n\rho_n) \\
s_1^{(2)} - (y_1^{(2)} + y_2^{(2)}) &= \Omega_P(n\rho_n), \quad x_1^{(2)} - (y_1^{(2)} + y_2^{(2)}) = \Omega_P(n\rho_n).
\end{aligned} \tag{B.35}$$

Now using the update for $u_3^{(2)}$, and defining the noise in the same way as in the first meta iteration,

$$\begin{aligned}
\langle u_3^{(2)}, v_2 \rangle &= \sum_{i \in C_{11}} \frac{2e^{8t(y_1^{(2)}+r_i^{(5)})} + e^{4t(x_1^{(2)}+r_i^{(4)})} + e^{4t(s_1^{(2)}+r_i^{(3)})}}{1 + e^{4t(s_1^{(2)}+r_i^{(3)})} + e^{4t(x_1^{(2)}+r_i^{(4)})} + e^{8t(y_1^{(2)}+r_i^{(5)})}} \\
&\quad + \sum_{i \in C_{12}} \frac{e^{4t(s_1^{(2)}+r_i^{(3)})} - e^{4t(x_2^{(2)}+r_i^{(4)})}}{1 + e^{4t(y_1^{(2)}+y_2^{(2)}+r_i^{(5)})} + e^{4t(s_1^{(2)}+r_i^{(3)})} + e^{4t(x_2^{(2)}+r_i^{(4)})}} \\
&\quad + \sum_{i \in C_{21}} \frac{e^{4t(x_1^{(2)}+r_i^{(4)})} - e^{4t(s_2^{(2)}+r_i^{(3)})}}{1 + e^{4t(y_1^{(2)}+y_2^{(2)}+r_i^{(5)})} + e^{4t(s_2^{(2)}+r_i^{(3)})} + e^{4t(x_1^{(2)}+r_i^{(4)})}} \\
&\quad - \sum_{i \in C_{22}} \frac{2e^{8t(y_2^{(2)}+r_i^{(5)})} + e^{4t(s_2^{(2)}+r_i^{(3)})} + e^{4t(x_2^{(2)}+r_i^{(4)})}}{1 + e^{4t(s_2^{(2)}+r_i^{(3)})} + e^{4t(x_2^{(2)}+r_i^{(4)})} + e^{8t(y_2^{(2)}+r_i^{(5)})}} \\
&\geq \sum_{i \in C_{11}} \frac{2e^{8t(y_1^{(2)}+r_i^{(5)})}}{1 + e^{4t(s_1^{(2)}+r_i^{(3)})} + e^{4t(x_1^{(2)}+r_i^{(4)})} + e^{8t(y_1^{(2)}+r_i^{(5)})}} \\
&\quad + \sum_{i \in C_{12}} \frac{e^{4t(s_1^{(2)}+r_i^{(3)})}}{1 + e^{4t(y_1^{(2)}+y_2^{(2)}+r_i^{(5)})} + e^{4t(s_1^{(2)}+r_i^{(3)})} + e^{4t(x_2^{(2)}+r_i^{(4)})}} \\
&\quad + \sum_{i \in C_{21}} \frac{e^{4t(x_1^{(2)}+r_i^{(4)})}}{1 + e^{4t(y_1^{(2)}+y_2^{(2)}+r_i^{(5)})} + e^{4t(s_2^{(2)}+r_i^{(3)})} + e^{4t(x_1^{(2)}+r_i^{(4)})}} \\
&\quad - n \exp(-\Omega_P(n\rho_n)) \\
&\quad \geq \frac{n}{2} - n \exp(-\Omega_P(n\rho_n)),
\end{aligned}$$

using the conditions (B.31) (B.32) (B.35) and Lemma 4. Since $\|u - z^*\|_1 = m - |\langle u, v_2 \rangle|$, $||u_3^{(2)} - z^*||_1 = n \exp(-\Omega_P(n\rho_n))$ after the second meta iteration.

Finally we show the later iterations conserve strong consistency. Since

$$\begin{aligned}\langle u_3^{(2)}, \mathbf{1}\rangle - m =& \sum_{i\in C_{11}} \frac{e^{8t(y_1^{(2)}+r_i^{(5)})}-1}{1+e^{4t(s_1^{(2)}+r_i^{(3)})}+e^{4t(x_1^{(2)}+r_i^{(4)})}+e^{8t(y_1^{(2)}+r_i^{(5)})}} \\ &+ \sum_{i\in C_{12}} \frac{e^{4t(y_1^{(2)}+y_2^{(2)}+r_i^{(5)})}-1}{1+e^{4t(y_1^{(2)}+y_2^{(2)}+r_i^{(5)})}+e^{4t(s_1^{(2)}+r_i^{(3)})}+e^{4t(x_2^{(2)}+r_i^{(4)})}} \\ &+ \sum_{i\in C_{21}} \frac{e^{4t(y_1^{(2)}+y_2^{(2)}+r_i^{(5)})}-1}{1+e^{4t(y_1^{(2)}+y_2^{(2)}+r_i^{(5)})}+e^{4t(s_2^{(2)}+r_i^{(3)})}+e^{4t(x_1^{(2)}+r_i^{(4)})}} \\ &+ \sum_{i\in C_{22}} \frac{e^{8t(y_2^{(2)}+r_i^{(5)})}-1}{1+e^{4t(s_2^{(2)}+r_i^{(3)})}+e^{4t(x_2^{(2)}+r_i^{(4)})}+e^{8t(y_2^{(2)}+r_i^{(5)})}} \\ =& n\exp(-\Omega_P(n\rho_n))\end{aligned}$$

by (B.31) (B.32) (B.35) and Lemma 4, we have

$$\begin{aligned}s_1^{(3)} =& a(\langle u_3^{(2)}, \mathbf{1}\rangle - m) + b\langle u_3^{(2)}, v_2\rangle = \frac{p-q}{4}n + n\rho_n \exp(-\Omega_P(n\rho_n)), \\ s_2^{(3)} =& a(\langle u_3^{(2)}, \mathbf{1}\rangle - m) - b\langle u_3^{(2)}, v_2\rangle = -\frac{p-q}{4}n + n\rho_n \exp(-\Omega_P(n\rho_n)).\end{aligned}$$

Next we note the noise in this iteration now arises from the whole graph A, and can be bounded by

$$\begin{aligned}r_i^{(7)} &= [R^{zz}, R^{zy}]_{i,\cdot}(u_3^{(2)} - \frac{1}{2}\mathbf{1}_n) \\ &= [R^{zz}, R^{zy}]_{i,\cdot}(u_3^{(2)} - z^*) + [R^{zz}, R^{zy}]_{i,\cdot}(z^* - \frac{1}{2}\mathbf{1}_n),\end{aligned}$$

where the second term is $O_P(\sqrt{n\rho_n \log n})$ uniformly for all i, applying Lemma 4. To bound the first term, note that

$$\begin{aligned}\max_i |[R^{zz}, R^{zy}]_{i,\cdot}(u_3^{(2)} - z^*)| &\le \|[R^{zz}, R^{zy}](u_3^{(2)} - z^*)\|_2 \\ &\le O_P(\sqrt{n\rho_n})\|u_3^{(2)} - z^*\|_1 = o_P(1).\end{aligned}$$

Therefore $r_i^{(7)}$ is uniformly $O_P(\sqrt{n\rho_n \log n})$ for all i. By a similar calculation to (B.34), we can check that condition (B.32) holds for $y_1^{(2)}$ and $s_1^{(3)}$, since when $s_1, x_1, y_1 = \Omega(n\rho_n)$ condition (B.32) and $1 - o_P(1) \leq \alpha_i(s_1, x_1, y_1) \leq 2 + o_P(1)$ guarantees each other and condition (B.32) is true in the previous iteration. We can check that condition (B.35) also holds. The rest of the argument can be applied to show $\|u_1^{(3)} - z^*\|_1 = n \exp(-\Omega_P(n\rho_n))$. At this point, all the arguments can be repeated for later iterations.

□

Proof of Corollary 1. We first consider $\mu > 0.5$. By (B.12), $s_1 = \Omega_P(n\rho_n)$, $s_2 = \Omega_P(n\rho_n)$. Since $r_i^{(0)} = O_P(\sqrt{n\rho_n \log n})$ uniformly for all i by Lemma 4, we have

$$\phi_i^{(1)} = \frac{1 + e^{4t(s_1 + r_i^{(0)})}}{3 + e^{4t(s_1 + r_i^{(0)})}} = 1 - \exp(-\Omega_P(n\rho_n))$$

for $i \in C_1$. Similarly for $i \in C_2$, and $\xi_i^{(1)} = \exp(-\Omega_P(n\rho_n))$. Define $u'_i = \mathbf{1}_{[i \in P_1]} + \mathbf{1}_{[i \in P_2]}$. Since the partition into P_1 and P_2 is random, $u'_i \sim$ iid Bernoulli(1/2), and $\|u_1 - u'\|_2 = \sqrt{n} \exp(-\Omega_P(n\rho_n))$.

In the second iteration, we can write

$$\begin{aligned}
\theta^{01} &= 4t([A^{yz}, A^{yy}] - \lambda J)(u_1 - \tfrac{1}{2}\mathbf{1}) \\
&= 4t([A^{yz}, A^{yy}] - \lambda J)(u_1 - u') + 4t([A^{yz}, A^{yy}] - \lambda J)(u' - \tfrac{1}{2}\mathbf{1}) \\
&= O_P(n\sqrt{\rho} \exp(-\Omega_P(n\rho_n))) + 4t([A^{yz}, A^{yy}] - \lambda J)(u' - \tfrac{1}{2}\mathbf{1}).
\end{aligned}$$

The signal part of the second term is $4t(x_1 \mathbf{1}_{C'_1} + x_2 \mathbf{1}_{C'_2})$ with x_1 and x_2 having the form of (B.12), with $u^{(0)}$ replaced by u'. Since $x_1, x_2 = \Omega_P(\sqrt{n}\rho_n)$, the

rest of the analysis proceeds like that of Theorem 2 restarting from the first iteration.

If $\mu < 0.5$, $s_1 = -\Omega_P(n\rho_n)$, $s_2 = -\Omega_P(n\rho_n)$. We have $\phi_i^{(1)} = \frac{1}{3} + \exp(-\Omega_P(n\rho_n))$, $\xi_i^{(1)} = \frac{2}{3} - \exp(-\Omega_P(n\rho_n))$. This time let $u' = \frac{1}{3}\mathbf{1}_{[i\in P_1]} + \frac{2}{3}\mathbf{1}_{[i\in P_2]}$, then θ^{01} can be written as

$$\theta^{01} = O_P(n\sqrt{\rho}\exp(-\Omega_P(n\rho_n))) + \frac{4t}{3}([A^{yz}, A^{yy}] - \lambda J)(3u' - \frac{3}{2}\mathbf{1}).$$

Noting that $3u_i' - 1 \sim$ iid Bernoulli(1/2), the same argument applies. □

Proof of Proposition 4. (i) We show each point is a stationary point by checking the vector update form of (3.9), (3.8), (3.10). Similar to Theorem 2, we have

$$\theta^{10} = 4t(s_1\mathbf{1}_{C_1} + s_2\mathbf{1}_{C_2} + r_i^{(0)})$$

where $r_i^{(0)} = O_P(\sqrt{n\rho_n \log n})$. Plugging $u^{(0)} = \mathbf{1}_n$ in (3.8), $s_1 = s_2 = 0.5(\frac{p+q}{2} - \lambda)n$. Similarly

$$\theta^{01} = 4t(x_1\mathbf{1}_{C_1} + x_2\mathbf{1}_{C_2} + r_i^{(1)}),\ \theta^{11} = 4t(y_1\mathbf{1}_{C_1} + y_2\mathbf{1}_{C_2} + r_i^{(1)})$$

where $x_1 = x_2 = 0.5(\frac{p+q}{2} - \lambda)n$, $y_1 = y_2 = (\frac{p+q}{2} - \lambda)n$. Plugging in (3.11) with $\frac{p+q}{2} - \lambda = \Omega_P(\rho_n)$ by Lemma 1, we have

$$\phi_i^{(1)} = 1 - \exp(-\Omega_P(n\rho_n)), \quad \xi_i^{(1)} = 1 - \exp(-\Omega_P(n\rho_n))$$

for all $i \in [m]$. Hence for sufficiently large n, $u^{(0)} = \mathbf{1}_n$ is the stationary point. For $u^{(0)} = \mathbf{0}_n$, similarly we have

$$\phi_i^{(1)} = \exp(-\Omega_P(n\rho_n)), \quad \xi_i^{(1)} = \exp(-\Omega_P(n\rho_n))$$

so $u^{(0)} = \mathbf{0}_n$ is also a stationary point for large n. (ii) The statement for $u^{(0)} = \mathbf{0}_n$ and $u^{(0)} = \mathbf{1}_n$ follows from Corollary 1 by $\mu = 0$ and $\mu = 1$. □

Proof of Proposition 5. Let $\hat{t}, \hat{\lambda}$ be constants defined in terms of $\hat{p}, \hat{q}$. First we observe using $\hat{p}, \hat{q}$ only replaces t, λ with $\hat{t}, \hat{\lambda}$ everywhere in the updates of Algorithm 2. We can check the analysis in Theorem 2 remains unchanged as long as

$$\text{i) } \frac{p+q}{2} > \hat{\lambda}, \quad \text{ii) } \hat{\lambda} - q = \Omega(\rho_n), \quad \text{iii) } \hat{t} = \Omega(1)$$

□

Proof of Theorem 3. Starting with $p^{(0)}$ and $q^{(0)}$ satisfying the conditions in Corollary 5, after two meta iterations of u updates, we have $\|u_3^{(2)} - z^*\|_1 = n\exp(-\Omega(n\rho_n))$. Updating $p^{(1)}, q^{(1)}$ with (3.13), we first analyze the population version of the numerator of $p^{(1)}$,

$$\begin{aligned}
&(\mathbf{1}_n - u)^T P(\mathbf{1}_n - u) + u^T P u + 2(\mathbf{1}_m - \psi^{10} - \psi^{01})^T \mathrm{diag}(P^{zy})\mathbf{1}_m \\
=&(\mathbf{1}_n - z^*)^T P(\mathbf{1}_n - z^*) + (z^*)^T P z^* - 2(u - z^*)^T P(\mathbf{1}_n - z^*) \\
&\quad + 2(z^*)^T P(u - z^*) + (u - z^*)^T P(u - z^*) + O(n\rho_n).
\end{aligned}$$

In the case of $u_3^{(2)}$, the above becomes

$$\frac{n^2}{2}p + O_P(n^{5/2}\rho_n \exp(-\Omega(n\rho_n))) + O(n\rho_n) = \frac{n^2}{2}p + O_P(n\rho_n).$$

Next we can rewrite the noise as

$$\begin{aligned}
&(\mathbf{1}_n - u)^T(A-P)(\mathbf{1}_n - u) + u^T(A-P)u \\
=&(\mathbf{1}_n - z^*)^T(A-P)(\mathbf{1}_n - z^*) - 2(u-z^*)^T(A-P)(\mathbf{1}_n - z^*) \\
&+ 2(z^*)^T(A-P)(u-z^*) + (u-z^*)^T(A-P)(u-z^*) + (z^*)^T(A-P)z^*.
\end{aligned}$$

Similarly in the case of $u_3^{(2)}$, the above is $O_P(\sqrt{n^2\rho_n})$. Therefore the numerator of $p^{(1)}$ is $\frac{n^2}{2}p + O_P(\sqrt{n^2\rho_n})$. To lower bound the denominator, note that

$$u^T(J-I)u + (\mathbf{1}-u)^T(J-I)(\mathbf{1}-u) \geq n^2/2 - 2n,$$

then we have $p^{(1)} = p + O_P(\sqrt{\rho_n}/n)$. The same analysis shows $q^{(1)} = q + O_P(\sqrt{\rho_n}/n)$.

Replacing p and q with $p^{(1)}$ and $q^{(1)}$ in the final analysis after the second meta iteration of Theorem 2 does not change the order of the convergence, and the rest of the arguments can be repeated. □

Appendix C

Appendix for Variational Inference with Discrete Latent Variables

C.1 The ARM Gradient Ascent Algorithm

We summarize the algorithm to compute ARM gradient for binary latent variables. Here we show the gradient with respect to the logits associated with the probability of Bernoulli random variables. If the logits are further generated by deterministic transform, the chain rule can be directly applied. For stochastic transforms, the implementation of ARM gradient is discussed in detail in Section 4.3 and summarized in Algorithm 4.

Algorithm 3 ARM gradient for a V-dimensional binary latent vector

input : Bernoulli distribution $\{q_{\phi_v}(z_v)\}_{v=1:V}$ with probability $\{\sigma(\phi_v)\}_{v=1:V}$, target $f(\boldsymbol{z})$; $\boldsymbol{z} = (z_1, \cdots, z_V)$, $\boldsymbol{\phi} = (\phi_1, \cdots, \phi_V)$

output : $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$ that maximize $\mathbb{E}(\boldsymbol{\phi}, \boldsymbol{\psi}) = \mathbb{E}_{\boldsymbol{z} \sim \prod_{v=1}^{V} q_{\phi_v}(z_v)}[f(\boldsymbol{z}; \boldsymbol{\psi})]$

Initialize $\boldsymbol{\phi}$, $\boldsymbol{\psi}$ randomly

while *not converged* **do**

- Sample a mini-batch of $\boldsymbol{x}$ from the data
- Sample $z_v \sim \text{Bernoulli}(\sigma(\phi_v))$ for $v = 1, \cdots, V$
- sample $u_v \sim \text{Uniform}(0, 1)$ for $v = 1, \cdots, V$, $\boldsymbol{u} = (u_1, \cdots, u_V)$
- $g_{\boldsymbol{\psi}} = \nabla_{\boldsymbol{\psi}} f(\boldsymbol{z}; \boldsymbol{\psi})$
- $f_\Delta(\boldsymbol{u}, \boldsymbol{\phi}) = f\big(\mathbf{1}_{[\boldsymbol{u} > \sigma(-\boldsymbol{\phi})]}\big) - f\big(\mathbf{1}_{[\boldsymbol{u} < \sigma(\boldsymbol{\phi})]}\big)$
- $g_{\boldsymbol{\phi}} = f_\Delta(\boldsymbol{u}, \boldsymbol{\phi})(\boldsymbol{u} - 0.5)$
- $\boldsymbol{\phi} = \boldsymbol{\phi} + \rho_t g_{\boldsymbol{\phi}}, \quad \boldsymbol{\psi} = \boldsymbol{\psi} + \eta_t g_{\boldsymbol{\psi}}, \qquad$ with stepsizes ρ_t, η_t

end

Algorithm 4 ARM gradient for a T-stochastic-hidden-layer binary network

input : Inference network $q_{\boldsymbol{w}_{1:T}}(\boldsymbol{b}_{1:T} \,|\, \boldsymbol{x}) = q_{\boldsymbol{w}_1}(\boldsymbol{b}_1 \,|\, \boldsymbol{x})\Big[\prod_{t=1}^{T-1} q_{\boldsymbol{w}_{t+1}}(\boldsymbol{b}_{t+1} \,|\, \boldsymbol{b}_t)\Big]$ where $q_{\boldsymbol{w}_t}(\boldsymbol{b}_t \,|\, \boldsymbol{b}_{t-1}) = \text{Bernoulli}(\boldsymbol{b}_t; \sigma(\mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1})))$, target $f(\boldsymbol{b}_{1:T}; \boldsymbol{\psi})$

output : $\boldsymbol{w}_{1:T}$ and $\boldsymbol{\psi}$ that maximize $\mathbb{E}(\boldsymbol{w}_{1:T}, \boldsymbol{\psi}) = \mathbb{E}_{\boldsymbol{b}_{1:T} \sim q_{\boldsymbol{w}_{1:T}}}[f(\boldsymbol{b}_{1:T}; \boldsymbol{\psi})]$

Initialize $\boldsymbol{w}_{1:T}$, $\boldsymbol{\psi}$ randomly
while *not converged* **do**
  Sample a mini-batch of $\boldsymbol{x}$ from data **for** *t = 1:T* **do**
    If $t \geq 2$, sample $\boldsymbol{b}_{t-1} \sim q(\boldsymbol{b}_{t-1}|\boldsymbol{b}_{t-2})$, if $t = 2, \boldsymbol{b}_{1:t-1} = \boldsymbol{b}_1$, else $\boldsymbol{b}_{1:t-1} = [\boldsymbol{b}_{1:t-2}, \boldsymbol{b}_{t-1}]$
    sample $\boldsymbol{u}_t \sim \prod \text{Uniform}(0, 1)$
    $\boldsymbol{b}_t^1 = \mathbf{1}_{[\boldsymbol{u}_t > \sigma(-\mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1}))]}$, $\boldsymbol{b}_t^2 = \mathbf{1}_{[\boldsymbol{u}_t < \sigma(\mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1}))]}$
    if $\boldsymbol{b}_t^1 = \boldsymbol{b}_t^2$ **then**
      $f_\Delta(\boldsymbol{b}_{1:t-1}, \boldsymbol{b}_{t:T}^1, \boldsymbol{b}_{t:T}^2) = 0$
    else
      $\boldsymbol{b}_{t+1:T}^1 \sim q(\boldsymbol{b}_{t+1:T}|\boldsymbol{b}_t^1)$, $\boldsymbol{b}_{t+1:T}^2 \sim q(\boldsymbol{b}_{t+1:T}|\boldsymbol{b}_t^2)$
      $f_\Delta(\boldsymbol{b}_{1:t-1}, \boldsymbol{b}_{t:T}^1, \boldsymbol{b}_{t:T}^2) = f(\boldsymbol{b}_{1:t-1}, \boldsymbol{b}_{t:T}^1) - f(\boldsymbol{b}_{1:t-1}, \boldsymbol{b}_{t:T}^2)$
      $g_{\boldsymbol{w}_t} = f_\Delta(\boldsymbol{b}_{1:t-1}, \boldsymbol{b}_{t:T}^1, \boldsymbol{b}_{t:T}^2)(\boldsymbol{u}_t - \frac{1}{2})^T \nabla_{\boldsymbol{w}_t} \mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1})$
    end
    $\boldsymbol{w}_t = \boldsymbol{w}_t + \rho_t g_{\boldsymbol{w}_t}$ with step-size ρ_t
  end
  $\boldsymbol{\psi} = \boldsymbol{\psi} + \eta_t \nabla_{\boldsymbol{\psi}} f(\boldsymbol{b}_{1:T}; \boldsymbol{\psi})$ with step-size η_t
end

C.2 Proofs of Main Results

Proof of Proposition 6. Since the gradients $g_{\text{ARM}}(u, \phi)$, $g_{\text{AR}}(u, \phi)$, and $g_{\text{R}}(z, \phi)$ are all unbiased, their expectations are the same as the true gradient $g_{true}(\phi) = \sigma(\phi)(1 - \sigma(\phi))[f(1) - f(0)]$. Denote $f_\Delta(u, \phi) = f(\mathbf{1}_{[u > \sigma(-\phi)]}) - f(\mathbf{1}_{[u < \sigma(\phi)]})$. Since

$$f_\Delta(u, \phi) = \begin{cases} 0, & \text{if } \sigma(-|\phi|) < u < \sigma(|\phi|), \\ f(1) - f(0), & \text{if } u > \sigma(|\phi|), \\ f(0) - f(1), & \text{if } u < \sigma(-|\phi|), \end{cases} \tag{C.1}$$

The second moment of $g_{\text{ARM}}(u,\phi)$ can be expressed as

$$\begin{aligned}
&\mathbb{E}_{u\sim\text{Uniform}(0,1)}[g^2_{\text{ARM}}(u,\phi)] = \mathbb{E}_{u\sim\text{Uniform}(0,1)}[f^2_\Delta(u,\phi)(u-1/2)^2] \\
=&\int_{\sigma(|\phi|)}^{1}[f(1)-f(0)]^2(u-1/2)^2du + \int_0^{\sigma(-|\phi|)}[f(0)-f(1)]^2(u-1/2)^2du \\
=&\frac{1}{12}[1-(\sigma(|\phi|)-\sigma(-|\phi|))^3][f(1)-f(0)]^2
\end{aligned}$$

Denoting $t=\sigma(|\phi|)-\sigma(-|\phi|)$, we can re-express $g_{true}(\phi)=\frac{1}{4}(1-t^2)[f(1)-f(0)]$. Thus, the variance of $g_{\text{ARM}}(u,\phi)$ can be expressed as

$$\begin{aligned}
\text{var}[g_{\text{ARM}}(u,\phi)] =&\frac{1}{4}\left[\frac{1}{3}(1-t^3)-\frac{1}{4}(1-t^2)^2\right][f(1)-f(0)]^2 \\
=&\frac{1}{16}(1-t)(t^3+\frac{7}{3}t^2+\frac{1}{3}t+\frac{1}{3})[f(1)-f(0)]^2 \qquad \text{(C.2)} \\
\leq&\frac{1}{25}[f(1)-f(0)]^2,
\end{aligned}$$

which reaches its maximum at $0.039788[f(1)-f(0)]^2$ when $t=\frac{\sqrt{5}-1}{2}$.

For the REINFORCE gradient, we have

$$\begin{aligned}
\mathbb{E}_{z\sim\text{Bernoulli}(\sigma(\phi))}[g^2_{\text{R}}(z,\phi)] =&\mathbb{E}_{z\sim\text{Bernoulli}(\sigma(\phi))}\left[f^2(z)(z(1-\sigma(\phi))-\sigma(\phi)(1-z))^2\right] \\
=&\sigma(\phi)(1-\sigma(\phi))[(1-\sigma(\phi))f^2(1)+\sigma(\phi)f^2(0)].
\end{aligned}$$

Therefore the variance can be expressed as

$$\begin{aligned}
&\text{var}[g_{\text{R}}(u,\phi)] \\
=&\sigma(\phi)(1-\sigma(\phi))\left[(1-\sigma(\phi))f^2(1)+\sigma(\phi)f^2(0)-\sigma(\phi)(1-\sigma(\phi))[f(1)-f(0)]^2\right] \\
=&\sigma(\phi)(1-\sigma(\phi))[(1-\sigma(\phi))f(1)+\sigma(\phi)f(0)]^2.
\end{aligned}$$

The largest variance satisfies

$$\sup_\phi \text{var}[g_{\text{R}}(z,\phi)] \geq \text{var}[g_{\text{R}}(z,0)] = \frac{1}{16}[f(1)+f(0)]^2,$$

and hence when f is always positive or negative, we have

$$\frac{\sup_\phi \mathrm{var}[g_{\mathrm{ARM}}(z,\phi)]}{\sup_\phi \mathrm{var}[g_{\mathrm{R}}(z,\phi)]} \leq \frac{16}{25}\big(1 - 2\frac{f(0)}{f(0)+f(1)}\big)^2 \leq \frac{16}{25}.$$

In summary, the ARM gradient has a variance that is bounded by $\frac{1}{25}(f(1) - f(0))^2$, and its worst-case variance is smaller than that of REINFORCE. $\square$

Proof of Proposition 7. We only need to prove for $K = 1$ and the proof for $K > 1$ automatically follows. Since

$$\mathbb{E}_{\boldsymbol{u}}[f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]})^2(u_v - 1/2)^2] = \mathbb{E}_{\boldsymbol{u}}[f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]})^2(u_v - 1/2)^2],$$

we have

$$\begin{aligned}
&\mathrm{var}(g_{\mathrm{ARM}_1,v}) - \mathrm{var}(g_{\mathrm{AR}_1,v}) \\
=& -3\mathbb{E}_{\boldsymbol{u}}[f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]})^2(u_v - 1/2)^2] + \mathbb{E}_{\boldsymbol{u}}[f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]})^2(u_v - 1/2)^2] \\
&- 2\mathbb{E}_{\boldsymbol{u}}[f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]})f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]})(u_v - 1/2)^2] \\
=& -\mathbb{E}_{\boldsymbol{u}}[f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]})^2(u_v - 1/2)^2] - \mathbb{E}_{\boldsymbol{u}}[f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]})^2(u_v - 1/2)^2] \\
&- 2\mathbb{E}_{\boldsymbol{u}}[f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]})f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]})(u_v - 1/2)^2] \\
=& -\mathbb{E}_{\boldsymbol{u}}\left[\big(f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]}) + f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]})\big)^2(u_v - 1/2)^2\right] \\
&\leq 0,
\end{aligned}$$

which shows that the estimation variance of $g_{\mathrm{ARM}_K,v}$ is guaranteed to be lower than that of the $g_{\mathrm{AR}_K,v}$, unless $f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]}) + f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]}) = 0$ almost surely.

Furthermore, since

$$
\begin{aligned}
&\mathrm{var}(g_{\mathrm{ARM}_1,v}) - \mathrm{var}(g_{\mathrm{AR}_2,v}) \\
=&\mathbb{E}_{\boldsymbol{u}}[(f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]}) - f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]}))^2(u_v - 1/2)^2] \\
&- \mathbb{E}_{\boldsymbol{u}^{(1)},\boldsymbol{u}^{(2)}}[(f(\mathbf{1}_{[\boldsymbol{u}^{(1)}<\sigma(\boldsymbol{\phi})]})(u_v^{(1)} - 1/2) + f(\mathbf{1}_{[\boldsymbol{u}^{(2)}<\sigma(\boldsymbol{\phi})]})(u_v^{(2)} - 1/2))^2] \\
=&-2\mathbb{E}_{\boldsymbol{u}^{(1)}}[f(\mathbf{1}_{[\boldsymbol{u}^{(1)}<\sigma(\boldsymbol{\phi})]})(u_v^{(1)} - 1/2)]\mathbb{E}_{\boldsymbol{u}^{(2)}}[f(\mathbf{1}_{[\boldsymbol{u}^{(2)}<\sigma(\boldsymbol{\phi})]})(u_v^{(2)} - 1/2))] \\
&- 2\mathbb{E}_{\boldsymbol{u}}[f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]})f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]}))(u_v - 1/2)^2] \\
=&-2\big(\mathbb{E}_{\boldsymbol{u}}[f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]})(u_v - 1/2)]\big)^2 \\
&- 2\mathbb{E}_{\boldsymbol{u}}[f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]})f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]}))(u_v - 1/2)^2],
\end{aligned}
$$

when f is always positive or negative, the variance of $g_{\mathrm{ARM}_K,v}$ is lower than that of $g_{\mathrm{AR}_{2K},v}$. □

Proof of Proposition 8. Denoting $g(\boldsymbol{u}) = g_{\mathrm{AR}}(\boldsymbol{u}) - \boldsymbol{b}(\boldsymbol{u})$, we have

$$
\mathrm{var}[g_v(\boldsymbol{u})] - \mathrm{var}[g_{\mathrm{AR},v}(\boldsymbol{u})] = -2\mathbb{E}_{\boldsymbol{u}}[g_{\mathrm{AR},v}(\boldsymbol{u})b_v(\boldsymbol{u})] + \mathbb{E}_{\boldsymbol{u}}[b_v^2(\boldsymbol{u})].
$$

To maximize the variance reduction, it is equivalant to consider the constrained optimization problem

$$
\begin{aligned}
\min_{b_v(\boldsymbol{u})} \quad & -2\mathbb{E}_{\boldsymbol{u}}[g_{\mathrm{AR},v}(\boldsymbol{u})b_v(\boldsymbol{u})] + \mathbb{E}_{\boldsymbol{u}}[b_v^2(\boldsymbol{u})] \\
\text{subject to:} \quad & b_v(\boldsymbol{u}) = -b_v(1-\boldsymbol{u}),
\end{aligned}
$$

which is the same as a Lagrangian problem as

$$
\begin{aligned}
&\min_{b_v(\boldsymbol{u}),\lambda_v(\boldsymbol{u})} \mathcal{L}(b_v(\boldsymbol{u}), \lambda_v(\boldsymbol{u})) \\
=&-2\mathbb{E}_{\boldsymbol{u}}[g_{\mathrm{AR},v}(\boldsymbol{u})b_v(\boldsymbol{u})] + \mathbb{E}_{\boldsymbol{u}}[b_v^2(\boldsymbol{u})] + \int \lambda_v(\boldsymbol{u})(b_v(\boldsymbol{u}) + b_v(1-\boldsymbol{u}))d\boldsymbol{u}.
\end{aligned}
$$

Setting $\frac{\delta\mathcal{L}}{\delta\lambda_v} = 0$ gives $b_v(\boldsymbol{u}) + b_v(1-\boldsymbol{u}) = 0$. By writing $\int \lambda_v(\boldsymbol{u})(b_v(\boldsymbol{u}) + b_v(1-\boldsymbol{u}))d\boldsymbol{u} = \int(\lambda_v(\boldsymbol{u}) + \lambda_v(1-\boldsymbol{u}))b_v(\boldsymbol{u})d\boldsymbol{u}$ and setting $\frac{\delta\mathcal{L}}{\delta b_v} = 0$, we have

$$[2g_{\mathrm{AR,v}}(\boldsymbol{u}) - 2b_v(\boldsymbol{u})]p(\boldsymbol{u}) = \lambda_v(\boldsymbol{u}) + \lambda_v(1-\boldsymbol{u}). \tag{C.3}$$

Interchange $\boldsymbol{u}$ and $1-\boldsymbol{u}$ gives

$$[2g_{\mathrm{AR,v}}(1-\boldsymbol{u}) - 2b_v(1-\boldsymbol{u})]p(1-\boldsymbol{u}) = \lambda_v(1-\boldsymbol{u}) + \lambda_v(\boldsymbol{u}). \tag{C.4}$$

Solving (C.3) and (C.4) with $b_v(\boldsymbol{u}) + b_v(1-\boldsymbol{u}) = 0$ and $p(\boldsymbol{u}) = p(1-\boldsymbol{u})$, we have the optimal baseline function as $b_v^*(\boldsymbol{u}) = \frac{1}{2}(g_{\mathrm{AR},v}(\boldsymbol{u}) - g_{\mathrm{AR},v}(1-\boldsymbol{u}))$. The proof is completed by noticing that $g_{\mathrm{AR}}(\boldsymbol{u}) - b^*(\boldsymbol{u})$ is the same as the single sample gradient estimate under the ARM estimator. □

Proof of Corollary 2. Since $b_v(\boldsymbol{u}) = c_v(1-2u)$ satisfies the anti-symmetric property, we can directly arrive at Corollary 2 using Proposition 8. Alternatively, since $\mathbb{E}_{\boldsymbol{u}}[f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]})^2(u_v - 1/2)^2] = \mathbb{E}_{\boldsymbol{u}}[f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]})^2(u_v - 1/2)^2]$ and $\mathbb{E}_{\boldsymbol{u}}[f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]})(u_v - 1/2)^2] = \mathbb{E}_{\boldsymbol{u}}[f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]})(u_v - 1/2)^2]$, for $g_{C,v} = (f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]}) - c_v)(1-2u_v)$, we have

$$\begin{aligned}
&\mathrm{var}(g_{C,v}) - \mathrm{var}(g_{\mathrm{ARM},v}) \\
=&\mathbb{E}_{\boldsymbol{u}}[(f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]}) - c_v)^2(1-2u_v)^2] - \mathbb{E}_{\boldsymbol{u}}[(f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]}) - f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]}))^2(u_v - 1/2)^2] \\
=&\mathbb{E}_{\boldsymbol{u}}\big[\big(4c_v^2 - 8c_v f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]}) + 2f^2(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]}) + 2f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]})f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]})\big)(u_v - 1/2)^2\big] \\
=&\mathbb{E}_{\boldsymbol{u}}\big[\big(f(\mathbf{1}_{[\boldsymbol{u}>\sigma(-\boldsymbol{\phi})]}) + f(\mathbf{1}_{[\boldsymbol{u}<\sigma(\boldsymbol{\phi})]}) - 2c_v\big)^2(u_v - 1/2)^2\big] \geq 0.
\end{aligned}$$

□

Proof of Proposition 9. First, the gradient with respect to $\boldsymbol{w}_1$ on $\mathcal{E}(\boldsymbol{w}_{1:T}) = \mathbb{E}_{q(\boldsymbol{b}_1)}\mathbb{E}_{q(\boldsymbol{b}_{2:T}\,|\,\boldsymbol{b}_1)}[f(\boldsymbol{b}_{1:T})]$, can be computed as

$$\nabla_{\boldsymbol{w}_1}\mathcal{E}(\boldsymbol{w}_{1:T}) = \mathbb{E}_{\boldsymbol{u}_1\sim\text{Uniform}(0,1)}[f_\Delta(\boldsymbol{u}_1, \mathcal{T}_{\boldsymbol{w}_1}(\boldsymbol{x}))(\boldsymbol{u}_1 - 1/2)]\nabla_{\boldsymbol{w}_1}\mathcal{T}_{\boldsymbol{w}_1}(\boldsymbol{x}),$$

$$\begin{aligned}\text{where } f_\Delta(\boldsymbol{u}_1, \mathcal{T}_{\boldsymbol{w}_1}(\boldsymbol{x})) = \mathbb{E}_{\boldsymbol{b}_{2:T}\sim q(\boldsymbol{b}_{2:T}\,|\,\boldsymbol{b}_1),\ \boldsymbol{b}_1=\mathbf{1}_{[\boldsymbol{u}_1>\sigma(-\mathcal{T}_{\boldsymbol{w}_1}(\boldsymbol{x}))]})[f(\boldsymbol{b}_{1:T})]\\ - \mathbb{E}_{\boldsymbol{b}_{2:T}\sim q(\boldsymbol{b}_{2:T}\,|\,\boldsymbol{b}_1),\ \boldsymbol{b}_1=\mathbf{1}_{[\boldsymbol{u}_1<\sigma(\mathcal{T}_{\boldsymbol{w}_1}(\boldsymbol{x}))]})[f(\boldsymbol{b}_{1:T})].\end{aligned}$$

Second, to compute the gradient with respect to $\boldsymbol{w}_t$, for $2 \le t \le T-1$,

$$\mathcal{E}(\boldsymbol{w}_{1:T}) = \mathbb{E}_{q(\boldsymbol{b}_{1:t-1})}\mathbb{E}_{q(\boldsymbol{b}_t\,|\,\boldsymbol{b}_{t-1})}\mathbb{E}_{q(\boldsymbol{b}_{t+1:T}\,|\,\boldsymbol{b}_t)}[f(\boldsymbol{b}_{1:T})],$$

the gradient is

$$\nabla_{\boldsymbol{w}_t}\mathcal{E}(\boldsymbol{w}_{1:T}) = \mathbb{E}_{q(\boldsymbol{b}_{1:t-1})}[\mathbb{E}_{\boldsymbol{u}_t}[f_\Delta(\boldsymbol{u}_t, \mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1}), \boldsymbol{b}_{1:t-1})(\boldsymbol{u}_t - 1/2)]\nabla_{\boldsymbol{w}_t}\mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1})]\,,$$

$$\begin{aligned}\text{with } f_\Delta(\boldsymbol{u}_t, \mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1}), \boldsymbol{b}_{1:t-1}) = \mathbb{E}_{\boldsymbol{b}_{t+1:T}\sim q(\boldsymbol{b}_{t+1:T}\,|\,\boldsymbol{b}_t),\ \boldsymbol{b}_t=\mathbf{1}_{[\boldsymbol{u}_t>\sigma(-\mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1}))]})[f(\boldsymbol{b}_{1:T})]\\ - \mathbb{E}_{\boldsymbol{b}_{t+1:T}\sim q(\boldsymbol{b}_{t+1:T}\,|\,\boldsymbol{b}_t),\ \boldsymbol{b}_t=\mathbf{1}_{[\boldsymbol{u}_t<\sigma(\mathcal{T}_{\boldsymbol{w}_t}(\boldsymbol{b}_{t-1}))]})[f(\boldsymbol{b}_{1:T})].\end{aligned}$$

Finally, to compute the gradient with respect to $\boldsymbol{w}_T$, we have

$$\nabla_{\boldsymbol{w}_T}\mathcal{E}(\boldsymbol{w}_{1:T}) = \mathbb{E}_{q(\boldsymbol{b}_{1:T-1})}\left[\mathbb{E}_{\boldsymbol{u}_T}[f_\Delta(\boldsymbol{u}_T, \mathcal{T}_{\boldsymbol{w}_T}(\boldsymbol{b}_{T-1}), \boldsymbol{b}_{1:T-1})(\boldsymbol{u}_T - 1/2)]\nabla_{\boldsymbol{w}_T}\mathcal{T}_{\boldsymbol{w}_T}(\boldsymbol{b}_{T-1})\right],$$

$$\begin{aligned}\text{where } f_\Delta(\boldsymbol{u}_T, \mathcal{T}_{\boldsymbol{w}_T}(\boldsymbol{b}_{T-1}), \boldsymbol{b}_{1:T-1}) = f(\boldsymbol{b}_{1:T-1}, \boldsymbol{b}_T = \mathbf{1}_{[\boldsymbol{u}_T>\sigma(-\mathcal{T}_{\boldsymbol{w}_T}(\boldsymbol{b}_{T-1}))]})\\ - f(\boldsymbol{b}_{1:T-1}, \boldsymbol{b}_T = \mathbf{1}_{[\boldsymbol{u}_T<\sigma(\mathcal{T}_{\boldsymbol{w}_T}(\boldsymbol{b}_{T-1}))]}).\end{aligned}$$

□

C.3 Additional Experimental Results

In Table C.1, we summarize the network structures for discrete VAE. The symbols "$\rightarrow$", "]",)", and "$\rightsquigarrow$" represent deterministic linear transform, leaky rectified linear units (LeakyReLU) [83] activation, sigmoid activation, and random samplin respectively, in the encoder; their reversed versions are used in the decoder.

Table C.1: The constructions of differently structured discrete variational auto-encoders.

	Nonlinear	Linear	Linear two layers
Encoder	784$\rightarrow$200]$\rightarrow$200]$\rightarrow$200)$\rightsquigarrow$200	784$\rightarrow$200)$\rightsquigarrow$200	784$\rightarrow$200)$\rightsquigarrow$200$\rightarrow$200)$\rightsquigarrow$200
Decoder	784$\leftsquigarrow$(784$\leftarrow$[200$\leftarrow$[200$\leftarrow$200	784$\leftsquigarrow$(784$\leftarrow$200	784$\leftsquigarrow$(784$\leftarrow$200$\leftsquigarrow$(200$\leftarrow$200

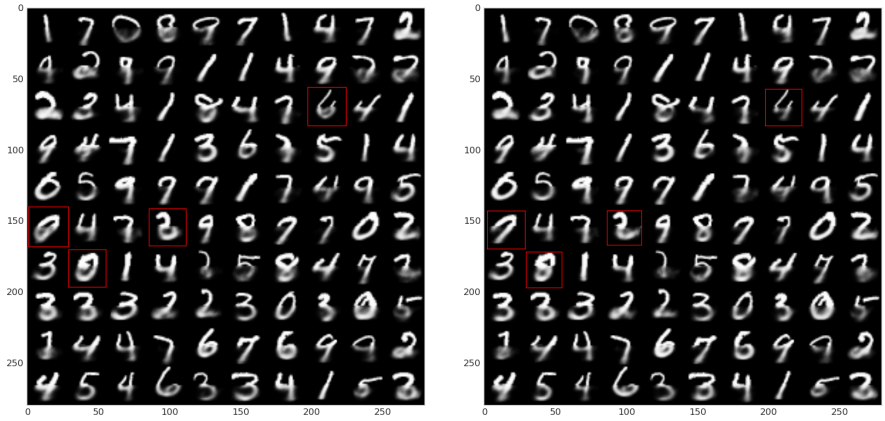


Figure C.1: Randomly selected example results of predicting the lower half of a MNIST digit given its upper half, using a binary stochastic network, which has two binary linear stochastic hidden layers, trained by the ARM estimator. Red squares highlight notable variations between two random draws.

Appendix D

Appendix for Meta-Learning with Variational Regularization

D.1 Algorithms for Meta Regularization

We present the detailed algorithm for meta-regularization on weights with conditional neural processes (CNP) in Algorithm 5 and with model-agnostic meta-learning (MAML) in Algorithm 6. For CNP, we add the regularization on the weights θ of encoder and leave other weights $\tilde{\theta}$ unrestricted. For MAML, we regularize the weights θ from input to an intermediate hidden layer and leave the weights $\tilde{\theta}$ for adaptation unregularized. In this way, we restrict the complexity of the pre-adaptation model not the post-adaptation model.

D.2 Meta Regularization on Activations

We show that $I(x^*; \hat{y}^*|z^*, \theta) \leq I(\hat{y}^*; \mathcal{D}|z^*, \theta)$. By Figure 5.2, we have that $I(\hat{y}^*; x^*|\theta, \mathcal{D}, z^*) = 0$. By the chain rule of mutual information we have

$$\begin{aligned} I(\hat{y}^*; \mathcal{D}|z^*, \theta) =& I(\hat{y}^*; \mathcal{D}|z^*, \theta) + I(\hat{y}^*; x^*|\mathcal{D}, \theta, z^*) \\ =& I(\hat{y}^*; x^*, \mathcal{D}|\theta, z^*) \\ =& I(x^*; \hat{y}^*|\theta, z^*) + I(\hat{y}^*; \mathcal{D}|\theta, z^*, x^*) \\ \geq& I(x^*; \hat{y}^*|\theta, z^*) \end{aligned} \tag{D.1}$$

Algorithm 5 Meta-Regularized CNP

input : Task distribution $p(\mathcal{T})$; Encoder weights distribution $q(\theta;\tau) = \mathcal{N}(\theta;\tau)$ with Gaussian parameters $\tau = (\theta_\mu, \theta_\sigma)$; Prior distribution $r(\theta)$ and Lagrangian multiplier β; $\tilde{\theta}$ that parameterizes feature extractor $h_{\tilde{\theta}}(\cdot)$ and decoder $T_{\tilde{\theta}}(\cdot)$. Stepsize α.

output : Network parameter τ, $\tilde{\theta}$.

Initialize τ, $\tilde{\theta}$ randomly

while *not converged* **do**

  Sample a mini-batch of $\{\mathcal{T}_i\}$ from $p(\mathcal{T})$

  Sample $\theta \sim q(\theta;\tau)$ with reparameterization **for** *all* $\mathcal{T}_i \in \{\mathcal{T}_i\}$ **do**

    Sample $\mathcal{D}_i = (\boldsymbol{x}_i, \boldsymbol{y}_i)$, $\mathcal{D}_i^* = (\boldsymbol{x}_i^*, \boldsymbol{y}_i^*)$ from $\mathcal{T}_i$

    Encode observation $\boldsymbol{z}_i = g_\theta(\boldsymbol{x}_i)$, $\boldsymbol{z}_i^* = g_\theta(\boldsymbol{x}_i^*)$

    Compute task context $\phi_i = a(h_{\tilde{\theta}}(\boldsymbol{z}_i, \boldsymbol{y}_i))$ with aggregator $a(\cdot)$

  end

  Update $\tilde{\theta} \leftarrow \tilde{\theta} + \alpha \nabla_{\tilde{\theta}} \sum_{\mathcal{T}_i} \log q(\boldsymbol{y}_i^* | T_{\tilde{\theta}}(\boldsymbol{z}_i^*, \phi_i))$

  Update $\tau \leftarrow \tau + \alpha \nabla_\tau [\sum_{\mathcal{T}_i} \log q(\boldsymbol{y}_i^* | T_{\tilde{\theta}}(\boldsymbol{z}_i^*, \phi_i)) - \beta D_{\mathbb{D}_{\mathrm{KL}}}(q(\theta;\tau)||r(\theta))]$

end

Algorithm 6 Meta-Regularized MAML

input : Task distribution $p(\mathcal{T})$; Weights distribution $q(\theta;\tau) = \mathcal{N}(\theta;\tau)$ with Gaussian parameters $\tau = (\theta_\mu, \theta_\sigma)$; Prior distribution $r(\theta)$ and Lagrangian multiplier β; Stepsize α, α'.

output : Network parameter τ, $\tilde{\theta}$.

Initialize τ, $\tilde{\theta}$ randomly **while** *not converged* **do**

  Sample a mini-batch of $\{\mathcal{T}_i\}$ from $p(\mathcal{T})$

  Sample $\theta \sim q(\theta;\tau)$ with reparameterization **for** *all* $\mathcal{T}_i \in \{\mathcal{T}_i\}$ **do**

    Sample $\mathcal{D}_i = (\boldsymbol{x}_i, \boldsymbol{y}_i)$, $\mathcal{D}_i^* = (\boldsymbol{x}_i^*, \boldsymbol{y}_i^*)$ from $\mathcal{T}_i$

    Encode observation $\boldsymbol{z}_i = g_\theta(\boldsymbol{x}_i)$, $\boldsymbol{z}_i^* = g_\theta(\boldsymbol{x}_i^*)$

    Compute task specific parameter $\phi_i = \tilde{\theta} + \alpha' \nabla_{\tilde{\theta}} \log q(\boldsymbol{y}_i | \boldsymbol{z}_i, \tilde{\theta})$

  end

  Update $\tilde{\theta} \leftarrow \tilde{\theta} + \alpha \nabla_{\tilde{\theta}} \sum_{\mathcal{T}_i} \log q(\boldsymbol{y}_i^* | \boldsymbol{z}_i^*, \phi_i)$

  Update $\tau \leftarrow \tau + \alpha \nabla_\tau [\sum_{\mathcal{T}_i} \log q(\boldsymbol{y}_i^* | \boldsymbol{z}_i^*, \phi_i) - \beta D_{\mathbb{D}_{\mathrm{KL}}}(q(\theta;\tau)||r(\theta))]$

end

Algorithm 7 Meta-Regularized Methods in Meta-testing

input : Meta-testing task $\mathcal{T}$ with training data $\mathcal{D} = (\boldsymbol{x}, \boldsymbol{y})$ and testing input $\boldsymbol{x}^*$, optimized parameters $\tau, \tilde{\theta}$.

output : Prediction $\hat{y}^*$

for *k from 1 to K* **do**

Sample $\theta_k \sim q(\theta; \tau)$ Encode observation $\boldsymbol{z}_k = g_{\theta_k}(\boldsymbol{x})$, $\boldsymbol{z}_k^* = g_{\theta_k}(\boldsymbol{x}^*)$ Compute task specific parameter $\phi_k = a(h_{\tilde{\theta}}(\boldsymbol{z}_k, \boldsymbol{y}))$ for MR-CNP and $\phi_k = \tilde{\theta} + \alpha' \nabla_{\tilde{\theta}} \log q(\boldsymbol{y}|\boldsymbol{z}_k, \tilde{\theta})$ for MR-MAML Predict $\hat{y}_k^* \sim q(\hat{y}^*|z_k^*, \phi_k, \tilde{\theta})$

end

Return prediction $\hat{y}^* = \frac{1}{K} \sum_{k=1}^{K} \hat{y}_k^*$

D.3 Meta Regularization on Weights

Similar to [1], we use ξ to denote the unknown parameters of the true data generating distribution. This defines a joint distribution $p(\xi, \mathcal{M}, \theta) = p(\xi)p(\mathcal{M}|\xi)q(\theta|\mathcal{M})$. Furthermore, we have a predictive distribution

$$q(\hat{y}^*|x^*, \mathcal{D}, \theta) = \mathbb{E}_{\phi|\theta,\mathcal{D}} \left[q(\hat{y}^*|x^*, \phi, \theta) \right].$$

The meta-training loss in Eq. 5.1 is an upper bound for the cross entropy $H_{p,q}(y_{1:N}^*|x_{1:N}^*, \mathcal{D}_{1:N}, \theta)$. Using an information decomposition of cross entropy [1], we have

$$\begin{aligned} & H_{p,q}(y_{1:N}^*|x_{1:N}^*, \mathcal{D}_{1:N}, \theta) && \text{(D.2)} \\ =& H(y_{1:N}^*|x_{1:N}^*, \mathcal{D}_{1:N}, \xi) + I(\xi; y_{1:N}^*|x_{1:N}^*, \mathcal{D}_{1:N}, \theta) \\ & + \mathbb{E}\left[D_{\mathcal{D}_{\mathrm{KL}}}(p(y_{1:N}^*|x_{1:N}^*, \mathcal{D}_{1:N}, \theta)||q(y_{1:N}^*|x_{1:N}^*, \mathcal{D}_{1:N}, \theta)) \right] \\ & + I(\mathcal{D}_{1:N}; \theta|x_{1:N}^*, \xi) - I(y_{1:N}^*, \mathcal{D}_{1:N}; \theta|x_{1:N}^*, \xi). && \text{(D.3)} \end{aligned}$$

Here the only negative term is the $I(y_{1:N}^*, \mathcal{D}_{1:N}; \theta|x_{1:N}^*, \xi)$, which quantifies the information that the meta-parameters contain about the meta-training

data beyond what can be inferred from the data generating parameters (i.e., memorization). Without proper regularization, the cross entropy loss can be minimized by maximizing this term. We can control its value by upper bounding it

$$\begin{aligned} I(y^*_{1:N}, \mathcal{D}_{1:N}; \theta | x^*_{1:N}, \xi) &= \mathbb{E}\left[\log \frac{q(\theta|\mathcal{M}, \xi)}{q(\theta|x^*_{1:N}, \xi)}\right] \\ &= \mathbb{E}\left[\log \frac{q(\theta|\mathcal{M})}{q(\theta|x^*_{1:N}, \xi)}\right] \\ &= \mathbb{E}\left[D_{\mathcal{D}_{\mathrm{KL}}}(q(\theta|\mathcal{M})||q(\theta|x^*_{1:N}, \xi))\right] \\ &\leq \mathbb{E}\left[D_{\mathcal{D}_{\mathrm{KL}}}(q(\theta|\mathcal{M})||r(\theta))\right], \end{aligned}$$

where the second equality follows because θ and ξ are conditionally independent given $\mathcal{M}$. This gives the regularization in Section 5.4.2.

D.4 Proof of the PAC-Bayes Generalization Bound

First, we prove a more general result and then specialize it. The goal of the meta-learner is to extract information about the meta-training tasks and the test task training data to serve as a prior for test examples from the novel task. This information will be in terms of a distribution Q over possible models. When learning a new task, the meta-learner uses the training task data $\mathcal{D}$ and a model parameterized by θ (sampled from $Q(\theta)$) and outputs a distribution $q(\phi|\mathcal{D}, \theta)$ over models. Our goal is to learn Q such that it performs well on novel tasks.

To formalize this, define

$$er(Q, \mathcal{D}, \mathcal{T}) = \mathbb{E}_{\theta \sim Q(\theta), \phi \sim q(\phi|\theta, \mathcal{D}), (x^*, y^*) \sim p(x, y|\mathcal{T})} \left[\mathcal{L}(\phi(x^*), y^*) \right] \tag{D.4}$$

where $\mathcal{L}(\phi(x^*), y^*)$ is a bounded loss in $[0, 1]$. Then, we would like to minimize the error on novel tasks

$$er(Q) = \min_{Q} \mathbb{E}_{\mathcal{T} \sim p(\mathcal{T}), \mathcal{D} \sim p(x, y|\mathcal{T})} \left[er(Q, \mathcal{D}, \mathcal{T}) \right] \tag{D.5}$$

Because we only have a finite training set, computing $er(Q)$ is intractable, but we can form an empirical estimate:

$$\hat{er}(Q, \mathcal{D}_1, \mathcal{D}_1^*, ..., \mathcal{D}_n, \mathcal{D}_n^*) = \frac{1}{n} \sum_{i=1}^{n} \underbrace{\mathbb{E}_{\theta \sim Q(\theta), \phi_i \sim q(\phi|\theta, \mathcal{D}_i)} \left[\frac{1}{K} \sum_{(x^*, y^*) \in \mathcal{D}_i^*} \mathcal{L}(\phi(x^*), y^*)) \right]}_{\hat{er}(Q, \mathcal{D}_i, \mathcal{D}_i^*)} \tag{D.6}$$

where for exposition we assume $K = |\mathcal{D}_i^*|$ is the same for all i. We would like to relate $er(Q)$ and $\hat{er}(Q, \mathcal{D}_1, \mathcal{D}_1^*, ..., \mathcal{D}_n, \mathcal{D}_n^*)$, but the challenge is that Q may depend on $\mathcal{D}_1, \mathcal{D}_1^*, ..., \mathcal{D}_n, \mathcal{D}_n^*$ due to the learning algorithm. There are two sources of generalization error: (i) error due to the finite number of observed tasks and (ii) error due to the finite number of examples observed per task. Closely following the arguments in [7], we apply a standard PAC-Bayes bound to each of these and combine the results with a union bound.

Theorem 6. *Let $Q(\theta)$ be a distribution over parameters θ and let $P(\theta)$ be a prior distribution. Then for any $\delta \in (0, 1]$, with probability at least $1 - \delta$, the following inequality holds uniformly for all distributions Q,*

$$er(Q) \leq \frac{1}{n} \sum_{i=1}^{n} \hat{er}(Q, \mathcal{D}_i, \mathcal{D}_i^*) + \left(\sqrt{\frac{1}{2(K-1)}} + \sqrt{\frac{1}{2(n-1)}} \right) \sqrt{(Q \| P) + \log \frac{n(K+1)}{\delta}} \tag{D.7}$$

Proof. To start, we state a classical PAC-Bayes bound and use it to derive generalization bounds on task and datapoint level generalization, respectively.

Theorem 7. *Let $\mathcal{X}$ be a sample space (i.e. a space of possible datapoints). Let $P(X)$ be a distribution over $\mathcal{X}$ (i.e. a data distribution). Let Θ be a hypothesis space. Given a "loss function" $l(\theta, X) : \Theta \times \mathcal{X} \to [0, 1]$ and a collection of M i.i.d. random variables sampled from $P(X)$, $X_1, ..., X_M$, let π be a prior distribution over hypotheses in Θ that does not depend on the samples but may depend on the data distribution $P(X)$. Then, for any $\delta \in (0, 1]$, the following bound holds uniformly for all posterior distributions ρ over Θ*

$$P\Big(\mathbb{E}_{X_i \sim P(X), \theta \sim \rho(\cdot)}\left[l(\theta, X_i)\right] \leq \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\theta \sim \rho(\cdot)}[l(\theta, X_m] + \sqrt{\frac{1}{2(M-1)}\left(D_{KL}(\rho\|\pi) + \log\frac{M}{\delta}\right)}, \forall\rho\Big) \geq 1 - \delta. \tag{D.8}$$

Meta-level generalization First, we bound the task-level generalization, that is we relate $er(Q)$ to $\frac{1}{n}\sum_{i=1}^{n} er(Q, \mathcal{D}_i, \mathcal{T}_i)$. Letting the samples be $X_i = (\mathcal{D}_i, \mathcal{T}_i)$, and $l(\theta, X_n) = \mathbb{E}_{\phi_i \sim q(\phi|\mathcal{D}_i, \theta), (x^*, y^*) \sim \mathcal{T}_i}[\mathcal{L}(\phi(x^*), y^*)]$, then Theorem 1 says that for any $\delta_0 \sim (0, 1]$

$$P\left(er(Q) \leq \frac{1}{n}\sum_{i=1}^{n} er(Q, \mathcal{D}_i, \mathcal{T}_i) + \sqrt{\frac{1}{2(n-1)}\left(\mathcal{D}_{\mathrm{KL}}(Q\|P) + \log\frac{n}{\delta_0}\right)}, \forall Q\right) \geq 1 - \delta_0, \tag{D.9}$$

where P is a prior over θ.

Within task generalization Next, we relate $er(Q, \mathcal{D}_i, \mathcal{T}_i)$ to $\hat{er}(Q, \mathcal{D}_i, \mathcal{D}_i^*)$ via the PAC-Bayes bound. For a fixed task i, task training data $\mathcal{D}_i$, a prior $\pi(\phi|\mathcal{T}_i)$ that only depends on the training data, and any $\delta_i \in (0, 1]$, we have

$$P\Big(\mathbb{E}_{(x^*, y^*) \sim p(x, y|\mathcal{T}_i)\rho(\phi_i)}\left[\mathcal{L}(\phi_i(x^*), y^*)\right] \leq \mathbb{E}_{\rho(\phi_i)}\left[\frac{1}{K}\sum_{(x^*, y^*) \in \mathcal{D}_i^*} \mathcal{L}(\phi_i(x^*), y^*)\right] + \sqrt{\frac{1}{2(K-1)}\left(\mathcal{D}_{\mathrm{KL}}(\rho||\pi) + \log\frac{K}{\delta_i}\right)}, \forall\rho\Big) \geq 1 - \delta_i.$$

Now, we choose $\pi(\phi|\mathcal{T}_i)$ to be $\int P(\theta)q(\phi|\theta,\mathcal{D}_i)d\theta$ and restrict $\rho(\phi)$ to be of the form $\int Q(\theta)q(\phi|\theta,\mathcal{D}_i)d\theta$ for any Q. While, π and ρ may be complicated distributions (especially, if they are defined implicitly), we know that with this choice of π and ρ, $\mathcal{D}_{\mathrm{KL}}(\rho||\pi) \leq \mathcal{D}_{\mathrm{KL}}(Q||P)$ [23], hence, we have

$$P\left(er(Q,\mathcal{D}_i,\mathcal{T}_i) \leq \hat{er}(Q,\mathcal{D}_i,\mathcal{D}_i^*) + \sqrt{\tfrac{1}{2(K-1)}\left(\mathcal{D}_{\mathrm{KL}}(Q\|P) + \log\tfrac{K}{\delta_i}\right)}, \forall Q\right) \geq 1-\delta_i \tag{D.10}$$

Overall bound on meta-learner generalization Combining Eq. (D.9) and (D.10) using the union bound, we have

$$\begin{aligned} P\Big(er(Q) \leq \tfrac{1}{n}\textstyle\sum_{i=1}^n \hat{er}(Q,\mathcal{D}_i,\mathcal{D}_i^*) + \sqrt{\tfrac{1}{2(K-1)}\mathcal{D}_{\mathrm{KL}}(Q\|P) + \log\tfrac{K}{\delta_i}} \\ + \sqrt{\tfrac{1}{2(n-1)}\mathcal{D}_{\mathrm{KL}}(Q\|P) + \log\tfrac{n}{\delta_0}}, \forall Q\Big) \geq 1 - (\textstyle\sum_i \delta_i + \delta_0) \end{aligned}$$

Choosing $\delta_0 = \frac{\delta}{K+1}$ and $\delta_i = \frac{K\delta}{n(K+1)}$, then we have:

$$\begin{aligned} &P\Big(er(Q) \leq \tfrac{1}{n}\textstyle\sum_{i=1}^n \hat{er}(Q,\mathcal{D}_i,\mathcal{D}_i^*) + \left(\sqrt{\tfrac{1}{2(K-1)}} + \sqrt{\tfrac{1}{2(n-1)}}\right)\sqrt{\mathcal{D}_{\mathrm{KL}}(Q\|P) + \log\tfrac{n(K+1)}{\delta}}, \forall Q\Big) \\ &\quad \geq 1-\delta. \end{aligned}$$

□

Because n is generally large, by Taylor expansion of the complexity term we have

$$\begin{aligned} &\left(\sqrt{\tfrac{1}{2(K-1)}} + \sqrt{\tfrac{1}{2(n-1)}}\right)\sqrt{\left(\mathcal{D}_{\mathrm{KL}}(Q||P) + \log\tfrac{n(K+1)}{\delta}\right)} \\ =&\tfrac{1}{2\sqrt{\log n(K+1)/\delta}}\left(\sqrt{\tfrac{1}{2(K-1)}} + \sqrt{\tfrac{1}{2(n-1)}}\right) \times \left(\mathcal{D}_{\mathrm{KL}}(Q||P) + 2\log(\tfrac{n(K+1)}{\delta})\right) + o(1) \end{aligned}$$

Re-defining the coefficient of KL term as β and omitting the constant and higher order term, we recover the meta-regularization bound in Eq.(5.4) when $Q(\theta) = \mathcal{N}(\theta;\theta_\mu,\theta_\sigma)$.

D.5 Experimental Details for Meta-Learning

D.5.1 Pose Prediction

We create a multi-task regression dataset based on the Pascal 3D data [156]. The dataset consists of 10 classes of 3D object such as "aeroplane", "sofa", "TV monitor", etc. Each class has multiple different objects and there are 65 objects in total. We randomly select 50 objects for meta-training and the other 15 objects for meta-testing. For each object, we use MuJoCo [144] to render 100 images with random orientations of the instance on a table, visualized in Figure 5.1. For the meta-learning algorithm, the observation (x) is the 128×128 gray-scale image and the label (y) is the orientation re-scaled to be within $[0, 10]$. For each task, we randomly sample 30 (x, y) pairs for an object and evenly split them between task training and task test data. We use a meta batch-size of 10 tasks per iteration.

For MR-CNP, we use a convolutional encoder with a fully connected bottom layer to map the input image to a 20-dimensional latent representation z and z^* for task training input x and test input x^*. The (z, y) are concatenated and mapped by the feature extractor and aggregator which are fully connected networks to the 200 dimensional task summary statistics ϕ. The decoder is a fully connected network that maps (ϕ, z^*) to the prediction $\hat{y}^*$.

For MR-MAML, we use a convolutional encoder to map the input image to a 14×14 dimensional latent representation z and z^*. The pairs (z, y) are used in the task adaptation step to get a task specific parameter ϕ via gradient descent. Then z^* is mapped to the prediction $\hat{y}^*$ with a convolutional predictor

parameterized by ϕ. The network is trained using 5 gradient steps with learning rate 0.01 in the inner loop for adaptation and evaluated using 20 gradient steps at the test-time.

D.5.2 Non-mutually-exclusive Classification

The Omniglot dataset consists of 20 instances of 1623 characters from 50 different alphabets. We randomly choose 1100 characters for meta-training and use the remaining for testing. The meta-training characters are partitioned into 60 disjoint sets for 20-way classification. The MiniImagenet dataset contains 100 classes of images including 64 training classes, 12 validation classes, and 24 test classes. We randomly partition the 64 meta-training classes into 13 disjoint sets for 5-way classification with one label having one less class of images than the others.

For MR-MAML we use a convolutional encoder similar to the pose prediction problem. The dimension of z and z^* is 14×14 for Omniglot and 20×20 for MiniImagenet. We use a convolutional decoder for both datasets. Following [32], we use a meta batch-size of 16 for 20-way Omniglot classification and meta batch-size of 4 for 5-way MiniImagenet classification. The meta-learning rate is chosen from $\{0.001, 0.005\}$ and the β for meta-regularized methods are chosen from $\{10^{-7}, 10^{-6}, \ldots, 10^{-3}\}$. The optimal hyperparameters are chosen for each method separately via cross-validation.

D.6 Additional Figures

We show a standard few-shot classification setup in meta-learning to illustrate a mutually-exclusive task distribution and a graphical model for the regularization on the activations.

Figure D.1: An example of *mutually-exclusive* task distributions.

In Figure D.2, D.3, we show the meta-test results on the non-mutually-exclusive sinusoid regression problem with CNP and MAML. For each row, the amplitudes of the true curves (orange) are randomly sampled uniformly from $[0.1, 4]$. For illustrative purposes, we fix the one-hot vector component of the input. In Figure D.2, the vanilla CNP cannot adapt to new task training data at test-time and the shape of prediction curve (blue) is determined by the one-hot amplitude not the task training data. Adding meta-regularization on both activation and weights enables the CNP to use the task training data at meta-training and causes the model to generalize well at test-time. In Figure D.3, due

to memorization, MAML adapts slowly and has large generalization error at test-time. Adding meta-regularization on both activation and weights recovers efficient adaptation.

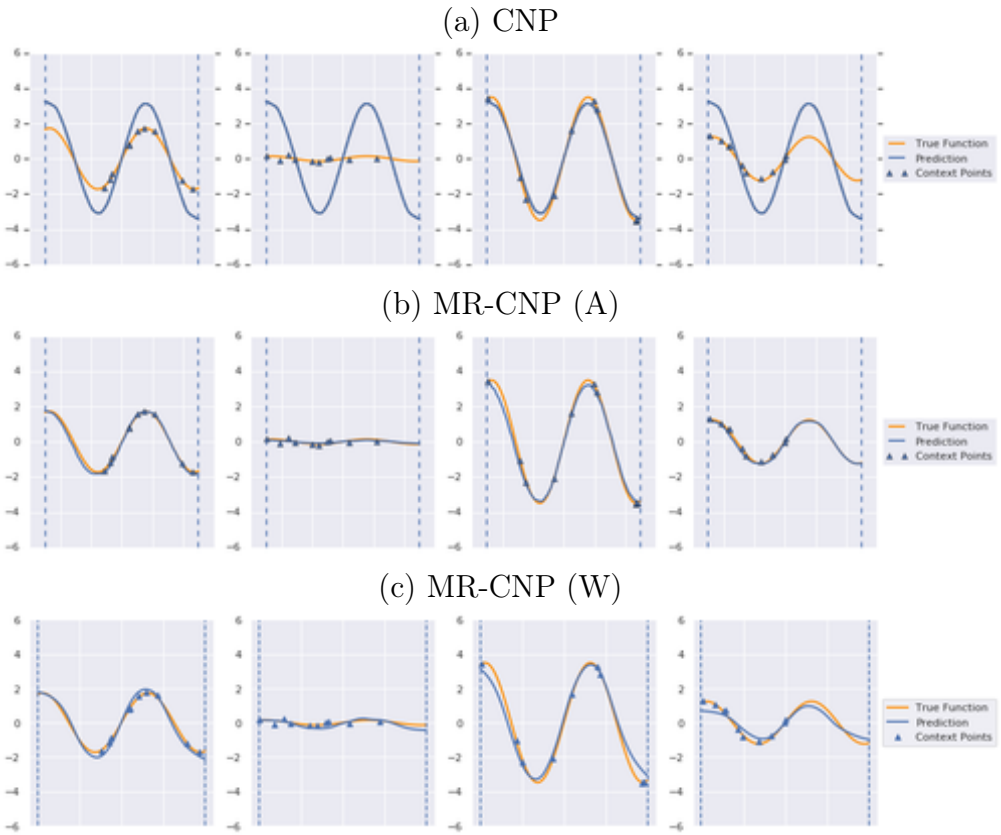
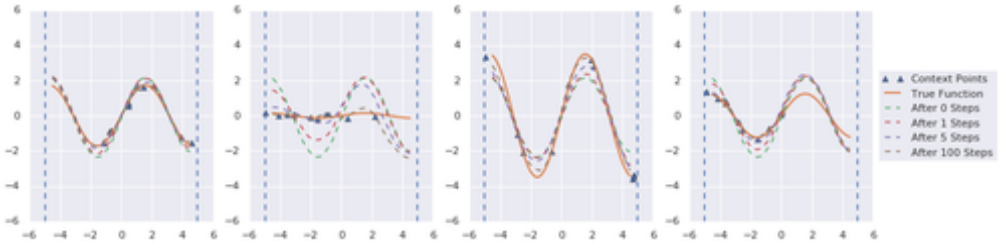
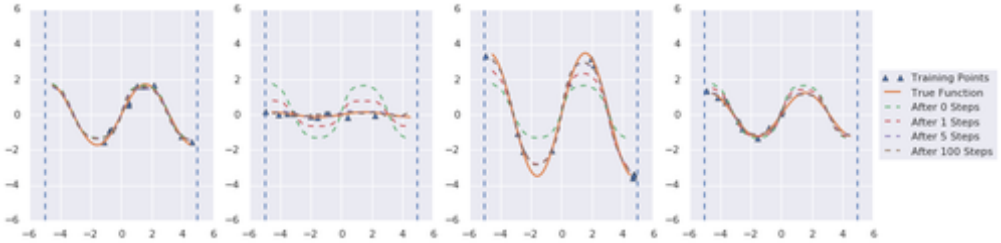


Figure D.2: Meta-test results on the non-mutually-exclusive sinusoid regression problem with CNP.

(a) MAML



(b) MR-MAML (A)



(c) MR-MAML (W)

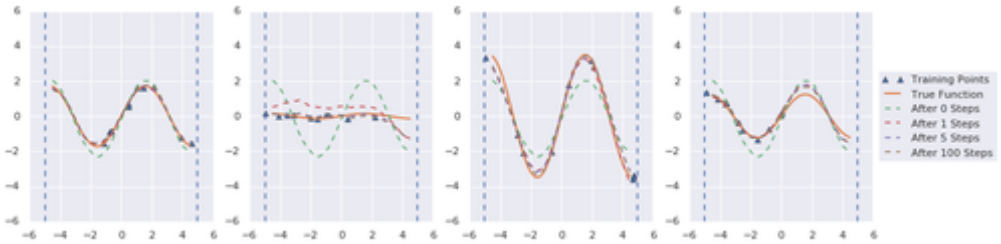


Figure D.3: Meta-test results on the non-mutually-exclusive sinusoid regression problem with MAML.

Bibliography

[1] Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. *The Journal of Machine Learning Research*, 19(1):1947–1980, 2018.

[2] F. V. Agakov and D. Barber. An auxiliary variational method. In *International Conference on Neural Information Processing*, 2004.

[3] Edoardo M Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. Mixed membership stochastic blockmodels. *Journal of machine learning research*, 9(Sep):1981–2014, 2008.

[4] Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Deep variational information bottleneck. In *ICLR*, 2017.

[5] A. A. Amini, A. Chen, P. J. Bickel, and E. Levina. Pseudo-likelihood methods for community detection in large sparse networks. *The Annals of Statistics*, 41(4):2097–2122, 2013.

[6] Arash A Amini, Elizaveta Levina, et al. On semidefinite relaxations for the block model. *The Annals of Statistics*, 46(1):149–179, 2018.

[7] Ron Amit and Ron Meir. Meta-learning by adjusting priors based on extended pac-bayes theory. In *International Conference on Machine Learning*, pages 205–214, 2018.

[8] Pranjal Awasthi and Andrej Risteski. On some provably correct cases of variational inference for topic models. In *Advances in Neural Information Processing Systems*, pages 2098–2106, 2015.

[9] Sivaraman Balakrishnan, Martin J Wainwright, Bin Yu, et al. Statistical guarantees for the em algorithm: From population to sample-based analysis. *The Annals of Statistics*, 45(1):77–120, 2017.

[10] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[11] P. J. Bickel and A. Chen. A nonparametric view of network models and newman–girvan and other modularities. *Proceedings of the National Academy of Sciences*, 106(50):21068–21073, 2009.

[12] Peter Bickel, David Choi, Xiangyu Chang, Hai Zhang, et al. Asymptotic normality of maximum likelihood and its variational approximation for stochastic blockmodels. *The Annals of Statistics*, 41(4):1922–1943, 2013.

[13] Christopher M Bishop. *Neural Networks for Pattern Recognition*. Oxford university press, 1995.

[14] Christopher M Bishop, Neil D Lawrence, Tommi Jaakkola, and Michael I Jordan. Approximating posterior distributions in belief networks using mixtures. In *NIPS*, pages 416–422, 1998.

[15] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

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

[17] Chester Ittner Bliss and Ronald A Fisher. Fitting the negative binomial distribution to biological data. *Biometrics*, 9(2):176–200, 1953.

[18] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pages 1613–1622, 2015.

[19] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

[20] Kamalika Chaudhuri, Fan Chung Graham, and Alexander Tsiatas. Spectral clustering of graphs with general degrees in the extended planted partition model. In *COLT*, volume 23 of *JMLR Proceedings*, pages 35.1–35.23. JMLR.org, 2012.

[21] Junyoung Chung, Sungjin Ahn, and Yoshua Bengio. Hierarchical multiscale recurrent neural networks. *arXiv preprint arXiv:1609.01704*, 2016.

[22] Amin Coja-Oghlan. Graph partitioning via adaptive spectral techniques. *Combinatorics, Probability and Computing*, 19(2):227–284, 2010.

[23] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[24] Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, 84(6): 066106, 2011.

[25] Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Phys. Rev. E*, Dec 2011.

[26] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[27] Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-linear independent components estimation. In *ICLR Workshop*, 2015.

[28] Harrison Edwards and Amos Storkey. Towards a neural statistician. *arXiv preprint arXiv:1606.02185*, 2016.

[29] Paul Erdös. On a lemma of littlewood and offord. *Bulletin of the American Mathematical Society*, 51(12):898–902, 1945.

[30] Xinjie Fan, Yizhe Zhang, Zhendong Wang, and Mingyuan Zhou. Adaptive correlated monte carlo for contextual categorical sequence generation. In *International Conference on Learning Representations*, 2020.

[31] Li Fei-Fei et al. A bayesian approach to unsupervised one-shot learning of object categories. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 1134–1141. IEEE, 2003.

[32] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org, 2017.

[33] Chelsea Finn, Kelvin Xu, and Sergey Levine. Probabilistic model-agnostic meta-learning. In *Advances in Neural Information Processing Systems*, pages 9516–9527, 2018.

[34] Michael C Fu. Gradient estimation. *Handbooks in operations research and management science*, 13:575–616, 2006.

[35] Tomer Galanti, Lior Wolf, and Tamir Hazan. A theoretical framework for deep transfer learning. *Information and Inference: A Journal of the IMA*, 5(2):159–209, 2016.

[36] Marta Garnelo, Dan Rosenbaum, Chris J Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo J Rezende, and SM Eslami. Conditional neural processes. *arXiv preprint*, 2018.

[37] Samuel J Gershman, Matthew D Hoffman, and David M Blei. Nonparametric variational inference. In *ICML*, pages 235–242, 2012.

[38] Behrooz Ghorbani, Hamid Javadi, and Andrea Montanari. An instability in variational inference for topic models. In *ICML*, 2019.

[39] Ryan J Giordano, Tamara Broderick, and Michael I Jordan. Linear response methods for accurate covariance estimates from mean field variational bayes. In *NIPS*, pages 1441–1449, 2015.

[40] Peter W Glynn. Likelihood ratio gradient estimation for stochastic systems. *Communications of the ACM*, 33(10):75–84, 1990.

[41] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[42] Prem K Gopalan and David M Blei. Efficient discovery of overlapping communities in massive networks. *Proceedings of the National Academy of Sciences*, 110(36):14534–14539, 2013.

[43] Jonathan Gordon, John Bronskill, Matthias Bauer, Sebastian Nowozin, and Richard E Turner. Meta-learning probabilistic inference for prediction. *arXiv preprint arXiv:1805.09921*, 2018.

[44] Erin Grant, Chelsea Finn, Sergey Levine, Trevor Darrell, and Thomas Griffiths. Recasting gradient-based meta-learning as hierarchical bayes. In *ICLR*, 2018.

[45] Will Grathwohl, Dami Choi, Yuhuai Wu, Geoff Roeder, and David Duvenaud. Backpropagation through the Void: Optimizing control variates for black-box gradient estimation. In *ICLR*, 2018.

[46] Karol Gregor, Ivo Danihelka, Andriy Mnih, Charles Blundell, and Daan Wierstra. Deep autoregressive networks. In *International Conference on Machine Learning*, 2014.

[47] Shixiang Gu, Sergey Levine, Ilya Sutskever, and Andriy Mnih. MuProp: Unbiased backpropagation for stochastic neural networks. In *ICLR*, 2016.

[48] Olivier Guédon and Roman Vershynin. Community detection in sparse networks via grothendieck's inequality. *Probability Theory and Related Fields*, 165(3-4):1025–1049, 2016.

[49] Simon Guiroy, Vikas Verma, and Christopher Pal. Towards understanding generalization in gradient-based meta-learning. *arXiv preprint arXiv:1907.07287*, 2019.

[50] Eissa D Habil. Double sequences and double series. *IUG Journal of Natural Studies*, 14(1), 2016.

[51] Jun Han, Fan Ding, Xianglong Liu, Lorenzo Torresani, Jian Peng, and Qiang Liu. Stein variational inference for discrete distributions. *arXiv preprint arXiv:2003.00605*, 2020.

[52] Shaobo Han, Xuejun Liao, David Dunson, and Lawrence Carin. Variational Gaussian copula inference. In *AISTATS*, pages 829–838, 2016.

[53] James Harrison, Apoorva Sharma, and Marco Pavone. Meta learning priors for efficient online bayesian regression. *arXiv preprint*, 2018.

[54] Geoffrey E Hinton, Peter Dayan, Brendan J Frey, and Radford M Neal. The "wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, 1995.

[55] Matthew Hoffman and David Blei. Stochastic structured variational inference. In *AISTATS*, pages 361–369, 2015.

[56] Matthew D Hoffman and David M Blei. Structured stochastic variational inference. In *Artificial Intelligence and Statistics*, 2015.

[57] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

[58] Paul W. Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

[59] Ferenc Huszár. Variational inference using implicit distributions. *arXiv preprint arXiv:1702.08235*, 2017.

[60] Daniel Jiwoong Im, Sungjin Ahn, Roland Memisevic, Yoshua Bengio, et al. Denoising criterion for variational auto-encoding framework. In *AAAI*, pages 2059–2065, 2017.

[61] T Jaakkola. Tutorial on variational approximation methods. *Advanced mean field methods: theory and practice*, page 129, 2001.

[62] Tommi S. Jaakkola and Michael I. Jordon. Improving the mean field approximation via the use of mixture distributions. In *Learning in Graphical Models*, pages 163–173. MIT Press, Cambridge, MA, USA, 1999. ISBN 0-262-60032-3.

[63] Muhammad Abdullah Jamal and Guo-Jun Qi. Task agnostic meta-learning for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11719–11727, 2019.

[64] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-softmax. In *ICLR*, 2017.

[65] Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Mach. Learn.*, 37(2), November 1999. ISSN 0885-6125.

[66] Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[67] Taesup Kim, Jaesik Yoon, Ousmane Dia, Sungwoong Kim, Yoshua Bengio, and Sungjin Ahn. Bayesian model-agnostic meta-learning. *arXiv preprint arXiv:1806.03836*, 2018.

[68] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[69] Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. In *ICLR*, 2013.

[70] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Advances in neural information processing systems*, pages 4743–4751, 2016.

[71] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2, 2015.

[72] Anders Krogh and John A Hertz. A simple weight decay can improve generalization. In *Advances in neural information processing systems*, pages 950–957, 1992.

[73] Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M Blei. Automatic differentiation variational inference. *Journal of Machine Learning Research*, 18(14):1–45, 2017.

[74] Andrew J Kurdila and Michael Zabarankin. Convex functional analysis (systems and control: Foundations and applications). *Switzerland: Birkhäuser*, 2005.

[75] Jeongyeol Kwon and Constantine Caramanis. Global convergence of em algorithm for mixtures of two component linear regression. *arXiv preprint arXiv:1810.05752*, 2018.

[76] Brenden Lake, Ruslan Salakhutdinov, Jason Gross, and Joshua Tenenbaum. One shot learning of simple visual concepts. In *Proceedings of the annual meeting of the cognitive science society*, volume 33, 2011.

[77] Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 29–37, 2011.

[78] Yoonho Lee, Wonjae Kim, and Seungjin Choi. Discrete infomax codes for meta-learning. *arXiv preprint arXiv:1905.11656*, 2019.

[79] Yingzhen Li and Richard E Turner. Gradient estimators for implicit models. In *ICLR*, 2018.

[80] Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Advances in neural information processing systems*, pages 2378–2386, 2016.

[81] Yucen Luo, Tian Tian, Jiaxin Shi, Jun Zhu, and Bo Zhang. Semi-crowdsourced clustering with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3212–3222, 2018.

[82] Lars Maaløe, Casper Kaae Sønderby, Søren Kaae Sønderby, and Ole Winther. Auxiliary deep generative models. In *ICML*, 2016.

[83] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, 2013.

[84] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *ICLR*, 2017.

[85] Catherine Matias and Vincent Miele. Statistical clustering of temporal networks through a dynamic stochastic block model. *Journal of the Royal Statistical Society: Series B*, 79(4):1119–1141, 2017.

[86] David A McAllester. Pac-bayesian model averaging. In *COLT*, volume 99, pages 164–170. Citeseer, 1999.

[87] Frank McSherry. Spectral partitioning of random graphs. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 529–537. IEEE, 2001.

[88] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Adversarial variational Bayes: Unifying variational autoencoders and generative adversarial networks. In *ICML*, 2017.

[89] Andrew C. Miller, Nicholas J. Foti, and Ryan P. Adams. Variational boosting: Iteratively refining posterior approximations. In *ICML*, pages 2420–2429, 2017.

[90] Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. In *ICML*, pages 1791–1799, 2014.

[91] Andriy Mnih and Danilo J Rezende. Variational inference for Monte Carlo objectives. *arXiv preprint arXiv:1602.06725*, 2016.

[92] Shakir Mohamed and Balaji Lakshminarayanan. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*, 2016.

[93] Dmitry Molchanov, Valery Kharitonov, Artem Sobolev, and Dmitry Vetrov. Doubly semi-implicit variational inference. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2593–2602, 2019.

[94] Elchanan Mossel, Joe Neeman, Allan Sly, et al. Belief propagation, robust reconstruction and optimal recovery of block models. *The Annals of Applied Probability*, 26(4):2211–2256, 2016.

[95] Soumendu Sundar Mukherjee, Purnamrita Sarkar, YX Rachel Wang, and Bowei Yan. Mean field for the stochastic blockmodel: Optimization landscape and convergence issues. In *Advances in Neural Information Processing Systems*, pages 10694–10704, 2018.

[96] Christian Naesseth, Francisco Ruiz, Scott Linderman, and David Blei. Reparameterization gradients through acceptance-rejection sampling algorithms. In *AISTATS*, pages 489–498, 2017.

[97] Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic,

real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*, 2018.

[98] Radford M Neal. Connectionist learning of belief networks. *Artificial Intelligence*, 56(1):71–113, 1992.

[99] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[100] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[101] Art B. Owen. *Monte Carlo Theory, Methods and Examples*, chapter 8 Variance Reduction. (n.p.), 2013.

[102] John Paisley, David M Blei, and Michael I Jordan. Variational Bayesian inference with stochastic search. In *ICML*, pages 1363–1370, 2012.

[103] George Papamakarios, Iain Murray, and Theo Pavlakou. Masked autoregressive flow for density estimation. In *NIPS*, pages 2335–2344, 2017.

[104] Debdeep Pati, Anirban Bhattacharya, and Yun Yang. On statistical optimality of variational bayes. *arXiv preprint arXiv:1712.08983*, 2017.

[105] Judea Pearl. *Reverend Bayes on inference engines: A distributed hierarchical approach.* Cognitive Systems Laboratory, School of Engineering and Applied Science, 1982.

[106] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier, 2014.

[107] Anastasia Pentina and Christoph Lampert. A pac-bayesian bound for lifelong learning. In *International Conference on Machine Learning*, pages 991–999, 2014.

[108] Amelia Perry and Alexander S Wein. A semidefinite program for unbalanced multisection in the stochastic block model. In *2017 International Conference on Sampling Theory and Applications (SampTA)*, pages 64–67. IEEE, 2017.

[109] Nicholas G Polson, James G Scott, and Jesse Windle. Bayesian inference for logistic models using Pólya–Gamma latent variables. *Journal of the American statistical Association*, 108(504):1339–1349, 2013.

[110] Tapani Raiko, Mathias Berglund, Guillaume Alain, and Laurent Dinh. Techniques for learning binary stochastic feedforward neural networks. *arXiv preprint arXiv:1406.2989*, 2014.

[111] Kate Rakelly, Aurick Zhou, Deirdre Quillen, Chelsea Finn, and Sergey Levine. Efficient off-policy meta-reinforcement learning via probabilistic context variables. *arXiv preprint arXiv:1903.08254*, 2019.

[112] Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *AISTATS*, pages 814–822, 2014.

[113] Rajesh Ranganath, Dustin Tran, and David Blei. Hierarchical variational models. In *ICML*, pages 324–333, 2016.

[114] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2016.

[115] Zahra S Razaee, Arash A Amini, and Jingyi Jessica Li. Matched bipartite block model with covariates. *Journal of Machine Learning Research*, 20 (34):1–44, 2019.

[116] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *ICML*, pages 1530–1538, 2015.

[117] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, pages 1278–1286, 2014.

[118] Karl Rohe, Sourav Chatterjee, Bin Yu, et al. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, 39(4): 1878–1915, 2011.

[119] Simone Romano, James Bailey, Vinh Nguyen, and Karin Verspoor. Standardized mutual information for clustering comparisons: one step further in adjustment for chance. In *International Conference on Machine Learning*, pages 1143–1151, 2014.

[120] Walter Rudin. *Principles of Mathematical Analysis*, volume 3. McGraw-hill New York, 1964.

[121] Ruslan Salakhutdinov and Iain Murray. On the quantitative analysis of deep belief networks. In *ICML*, pages 872–879, 2008.

[122] Tim Salimans and David A Knowles. Fixed-form variational posterior approximation through stochastic linear regression. *Bayesian Analysis*, 8 (4):837–882, 2013.

[123] Tim Salimans, Diederik Kingma, and Max Welling. Markov chain Monte Carlo and variational inference: Bridging the gap. In *ICML*, 2015.

[124] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850, 2016.

[125] Purnamrita Sarkar, Y. X. Rachel Wang, and Soumendu Sunder Mukherjee. When random initializations help: a study of variational inference for community detection. *arXiv e-prints*, art. arXiv:1905.06661, May 2019.

[126] Purnamrita Sarkar, YX Wang, and Soumendu Sunder Mukherjee. When random initializations help: a study of variational inference for community detection. *arXiv preprint arXiv:1905.06661*, 2019.

[127] Lawrence K Saul and Michael I Jordan. Exploiting tractable substructures in intractable networks. In *NIPS*, pages 486–492, 1996.

[128] Lawrence K Saul, Tommi Jaakkola, and Michael I Jordan. Mean field theory for sigmoid belief networks. *Journal of Artificial Intelligence Research*, 4:61–76, 1996.

[129] Jurgen Schmidhuber. Evolutionary principles in self-referential learning. *On learning how to learn: The meta-meta-... hook.) Diploma thesis, Institut f. Informatik, Tech. Univ. Munich*, 1:2, 1987.

[130] Jiaxin Shi, Shengyang Sun, and Jun Zhu. Implicit variational inference with kernel density ratio fitting. *arXiv preprint arXiv:1705.10119*, 2017.

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

[132] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder variational autoencoders. In *NIPS*, pages 3738–3746, 2016.

[133] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15 (1):1929–1958, 2014.

[134] Masashi Sugiyama, Taiji Suzuki, and Takafumi Kanamori. *Density ratio estimation in machine learning*. Cambridge University Press, 2012.

[135] Da Tang, Dawen Liang, Tony Jebara, and Nicholas Ruozzi. Correlated variational auto-encoders. In *International Conference on Machine Learning*, pages 6135–6144, 2019.

[136] Yichuan Tang and Ruslan R Salakhutdinov. Learning stochastic feedforward neural networks. In *NIPS*, pages 530–538, 2013.

[137] Yunhao Tang, Mingzhang Yin, and Mingyuan Zhou. Augment-reinforce-merge policy gradient for binary stochastic policy. *arXiv preprint arXiv:1903.05284*, 2019.

[138] Joshua Brett Tenenbaum. *A Bayesian framework for concept learning*. PhD thesis, Massachusetts Institute of Technology, 1999.

[139] Sebastian Thrun and Lorien Pratt. *Learning to learn*. Springer Science & Business Media, 2012.

[140] Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pages 1–5. IEEE, 2015.

[141] Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

[142] Michalis Titsias and Miguel Lázaro-Gredilla. Doubly stochastic variational bayes for non-conjugate inference. In *ICML*, pages 1971–1979, 2014.

[143] Michalis K Titsias and Miguel Lázaro-Gredilla. Local expectation gradients for black box variational inference. In *NIPS*, pages 2638–2646. MIT Press, 2015.

[144] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[145] Dustin Tran, David Blei, and Edo M Airoldi. Copula variational inference. In *NIPS*, pages 3564–3572, 2015.

[146] Dustin Tran, Rajesh Ranganath, and David Blei. Hierarchical implicit models and likelihood-free variational inference. In *NIPS*, pages 5529–5539, 2017.

[147] George Tucker, Andriy Mnih, Chris J Maddison, John Lawson, and Jascha Sohl-Dickstein. REBAR: Low-variance, unbiased gradient estimates for discrete latent variable models. In *NIPS*, pages 2624–2633, 2017.

[148] Masatoshi Uehara, Issei Sato, Masahiro Suzuki, Kotaro Nakayama, and Yutaka Matsuo. Generative adversarial nets from a density ratio estimation perspective. *arXiv preprint arXiv:1610.02920*, 2016.

[149] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *NIPS*, pages 6306–6315, 2017.

[150] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638, 2016.

[151] Martin J Wainwright, Michael I Jordan, et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2008.

[152] Bo Wang, DM Titterington, et al. Convergence properties of a general algorithm for calculating variational bayesian estimates for a normal mixture model. *Bayesian Analysis*, 1(3):625–650, 2006.

[153] Ted Westling and Tyler H. McCormick. Beyond prediction: A framework for inference with variational approximations in mixture models. *arXiv preprint arXiv:1510.08151*, 2015.

[154] Mateusz Wilinski, Piero Mazzarisi, Daniele Tantari, and Fabrizio Lillo. Detectability of macroscopic structures in directed asymmetric stochastic block model. *Physical Review E*, 99(4):042310, 2019.

[155] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Reinforcement Learning*, pages 5–32. Springer, 1992.

[156] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2014.

[157] Eric P Xing, Michael I Jordan, and Stuart Russell. A generalized mean field algorithm for variational inference in exponential families. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, pages 583–591. Morgan Kaufmann Publishers Inc., 2002.

[158] Ji Xu, Daniel J Hsu, and Arian Maleki. Global analysis of expectation maximization for mixtures of two gaussians. In *Advances in Neural Information Processing Systems*, pages 2676–2684, 2016.

[159] Bowei Yan, Mingzhang Yin, and Purnamrita Sarkar. Convergence of gradient em on multi-component mixture of gaussians. In *Advances in Neural Information Processing Systems*, pages 6956–6966, 2017.

[160] Xinyang Yi, Constantine Caramanis, and Sujay Sanghavi. Alternating minimization for mixed linear regression. In *International Conference on Machine Learning*, pages 613–621, 2014.

[161] Mingzhang Yin and Mingyuan Zhou. Semi-implicit variational inference. In *International Conference on Machine Learning*, pages 5646–5655, 2018.

[162] Mingzhang Yin and Mingyuan Zhou. ARM: Augment-REINFORCE-merge gradient for stochastic binary networks. In *International Conference on Learning Representations*, 2019.

[163] Mingzhang Yin and Mingyuan Zhou. Semi-implicit generative model. In *Workshop on Bayesian Deep Learning*, 2019.

[164] Mingzhang Yin, Yuguang Yue, and Mingyuan Zhou. ARSM: Augment-reinforce-swap-merge estimator for gradient backpropagation through categorical variables. In *International Conference on Machine Learning*, 2019.

[165] Mingzhang Yin, George Tucker, Mingyuan Zhou, Sergey Levine, and Chelsea Finn. Meta-learning without memorization. In *International Conference on Learning Representations*, 2020.

[166] Mingzhang Yin, YX Wang, and Purnamrita Sarkar. A theoretical case study of structured variational inference for community detection. In *International Conference on Artificial Intelligence and Statistics*, 2020.

[167] Yuguang Yue, Yunhao Tang, Mingzhang Yin, and Mingyuan Zhou. Discrete action on-policy learning with action-value critic. In *International Conference on Artificial Intelligence and Statistics*, 2020.

[168] Anderson Y Zhang and Harrison H Zhou. Theoretical and computational guarantees of mean field variational inference for community detection. *arXiv preprint arXiv:1710.11268*, 2017.

[169] Aonan Zhang and John Paisley. Markov mixed membership models. In *International Conference on Machine Learning*, pages 475–483, 2015.

[170] Ruixiang Zhang, Tong Che, Zoubin Ghahramani, Yoshua Bengio, and Yangqiu Song. Metagan: An adversarial approach to few-shot learning.

In *Advances in Neural Information Processing Systems*, pages 2365–2374, 2018.

[171] Mingyuan Zhou, Lingbo Li, David Dunson, and Lawrence Carin. Lognormal and gamma mixed negative binomial regression. In *ICML*, pages 859–866, 2012.

[172] Luisa M Zintgraf, Kyriacos Shiarlis, Vitaly Kurin, Katja Hofmann, and Shimon Whiteson. Fast context adaptation via meta-learning. In *International Conference on Machine Learning*, 2019.

Vita

Mingzhang Yin was born in Kunming, China, 1993, the son of Ying Hu and Cunzhen Yin. He spent most of his childhood on the campus of Kunming University of Science and Technology. He was guided to get in touch with different subjects in the sciences and liberal arts at an early age by his grandparents and parents.

Mingzhang earned a bachelor's degree in mathematics from Fudan University, Shanghai in 2015. He then moved to the United States to pursue a doctoral degree in statistics at the University of Texas at Austin. His doctoral research focuses on Bayesian statistics and machine learning, from both theoretical and methodological perspectives.

To continue the research in statistics, mathematics and machine learning, Mingzhang has accepted an offer as a postdoctoral fellow at the Data Science Institute, Columbia University, with an expected start date of July 1st, 2020.

Permanent email address: mzyin11@gmail.com
This dissertation was typed by the author.