Journal of Machine Learning Research 22 (2021) 1-44 Submitted 6/20; Revised 10/21; Published 11/21

Multilevel Monte Carlo Variational Inference

Masahiro Fujisawa FUJISAWA@MS.K.U-TOKYO.AC.JP
Graduate School of Frontier Sciences
The University of Tokyo
5-1-5 Kashiwanoha, Kashiwa, Chiba 277-8561, Japan
Center for Advanced Intelligence Project
RIKEN
1-4-1 Nihonbashi, Chuo-ku, Tokyo 103-0027, Japan

Issei Sato SATO@K.U-TOKYO.AC.JP
Graduate School of Information Science and Technology
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan

Editor: Erik Sudderth

Abstract

We propose a variance reduction framework for variational inference using the Multilevel Monte Carlo (MLMC) method. Our framework is built on reparameterized gradient estimators and "recycles" parameters obtained from past update history in optimization. In addition, our framework provides a new optimization algorithm based on stochastic gradient descent (SGD) that adaptively estimates the sample size used for gradient estimation according to the ratio of the gradient variance. We theoretically show that, with our method, the variance of the gradient estimator decreases as optimization proceeds and that a learning rate scheduler function helps improve the convergence. We also show that, in terms of the *signal-to-noise* ratio, our method can improve the quality of gradient estimation by the learning rate scheduler function without increasing the initial sample size. Finally, we confirm that our method achieves faster convergence and reduces the variance of the gradient estimator compared with other methods through experimental comparisons with baseline methods using several benchmark datasets.

Keywords: variational inference, variance reduction, approximate Bayesian inference, gradient estimation, stochastic optimization

1. Introduction

Variational inference (VI) (Jordan et al., 1999) has been successfully used in the context of approximate Bayesian inference. The object of VI is to seek a distribution from a variational family of distributions that best approximates an intractable posterior distribution (Miller et al., 2017). Unfortunately, the objective function of VI itself is often intractable and cannot be optimized in a closed form in modern complex models such as Bayesian neural networks. In this case, we often optimize the objective function based on the stochastic gradient estimated on the Monte Carlo samples from the variational distribution, which is called Monte Carlo variational inference (MCVI) or black box variational inference (Ranganath et al., 2014). However, the stochastic gradient obtained with Monte Carlo approximation causes a slow

©2021 Masahiro Fujisawa and Issei Sato.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v22/20-653.html.

convergence owing to the high variance. Therefore, the variance of the stochastic gradient estimator should be controlled carefully to make MCVI useful.

There are two standard MCVI gradient estimators: the score function gradient estimator (Paisley et al., 2012; Ranganath et al., 2014) and the reparameterized gradient estimator (Titsias and Lázaro-Gredilla, 2014; Rezende et al., 2014; Kingma and Welling, 2014). The score function gradient estimator is a general tool for MCVI, which can be applied to both discrete and continuous random variables; however, it often has high variance, and the training becomes difficult (Buchholz et al., 2018). In contrast, the reparameterized gradient estimator often has a lower variance for continuous random variables, although it is restricted to models where the variational family can be reparametrized via a differentiable mapping. Recently, there has been extensive research on making the reparameterized gradient estimator a more general tool, including a unified view of these two gradient estimators provided by Ruiz et al. (2016b). For example, Tokui and Sato (2016) and Jang et al. (2017) proposed a reparameterization trick for discrete or categorical variables. Furthermore, Tucker et al. (2017) proposed a low-variance reparameterized gradient estimation method for discrete latent variables. Grathwohl et al. (2018) presented a general method for unbiased gradient estimation of black-box functions and applied it to discrete variational inference and reinforcement learning. Moreover, the theoretical properties of the reparameterized gradient have been actively analyzed recently (Xu et al., 2019; Domke, 2019). Owing to these studies, the use of the reparameterized gradient has become a more practical way to reduce the variance of gradient estimators.

To reduce the variance of the score/reparameterized gradient estimators, we often use control variates (Glasserman, 2003; Miller et al., 2017; Geffner and Domke, 2020) or importance sampling (Ruiz et al., 2016a; Burda et al., 2016; Sakaya and Klami, 2017; Li et al., 2018). However, their use requires the construction of appropriate auxiliary functions to achieve effective variance reduction, and thus their performance depends on such a heuristic selection. Recently, Buchholz et al. (2018) have proposed a variance reduction scheme using the randomized quasi-Monte Carlo (RQMC) method for MCVI, which does not require the design of an auxiliary function and improves the order of the conventional gradient estimator variance, $\mathcal{O}(N^{-1})$, to $\mathcal{O}(N^{-2})$ in the best case. Although the directions proposed in this study are extremely important for making MCVI more practical, the advantage of the RQMC method is less than theoretically expected when the integration is not sufficiently smooth and/or its dimension is high (Morokoff and Caflisch, 1994). Furthermore, the performance of QMC-based methods sometimes becomes worse than that of MC methods owing to a potentially unfavorable interaction between the underlying deterministic points and the function to be estimated (Lemieux, 2009). Therefore, it is necessary to establish a *sampling-based* variance reduction method that achieves a stable performance against the problematic factors described above.

In this paper, we propose a novel *sampling-based* variance reduction framework for MCVI that "recycles" past gradients and parameters based on the Multilevel Monte Carlo (MLMC) method. Our method is naturally derived from the reparameterized gradient estimator and is also independent of the design of auxiliary functions for variance reduction. Furthermore, our method theoretically guarantees convergence acceleration and the quality of the gradient estimator. In addition, several experiments confirm that our method converges faster, reduces variance better, and sometimes achieves better prediction performance than the baseline

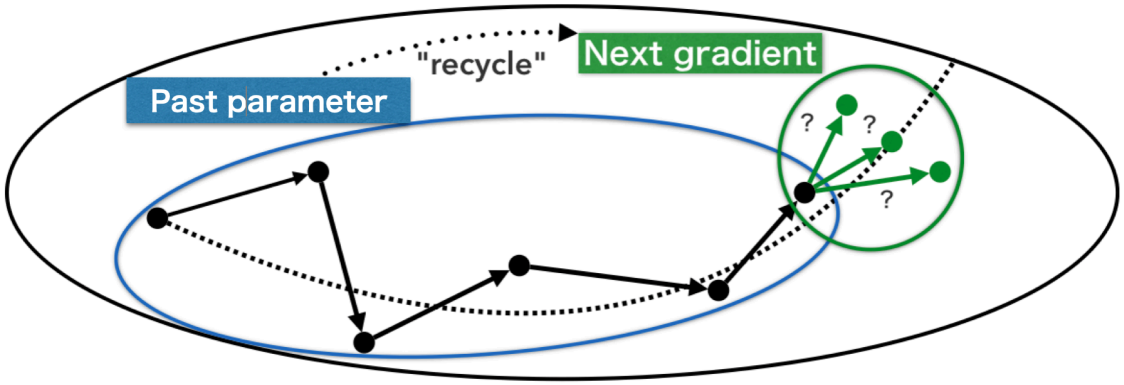

Figure 1: Concept of our method. In the context of traditional MCVI, the gradient from the current point (the rightmost point in the blue circle) is estimated with variance reduction techniques, and the parameters are updated (moved to the next green point in the green circle). On the other hand, our method uses the history of the parameters (all points in blue circles) to estimate the gradient with low variance and update the parameters.

methods. Since MLMC is a pure MC sampling method, our method can be combined with various variance reduction methods that have been proposed (Ranganath et al., 2014; Ruiz et al., 2016a; Burda et al., 2016; Sakaya and Klami, 2017; Miller et al., 2017; Li et al., 2018; Tucker et al., 2019), but it is unclear whether this is possible with RQMC-based methods (Buchholz et al., 2018). We note that our framework can be easily implemented in modern inference libraries such as Pytorch (Paszke et al., 2019). Our contributions are as follows.

- We investigate the idea of using the MLMC method for MCVI on reparameterized gradient estimation.
- On the basis of the above idea, we develop an algorithm that provides a low-variance gradient estimator with a small number of samples by "recycling" the parameters in the past (see Figure 1).
- Our convergence analysis shows that, in our method, the learning rate scheduler function helps accelerate the optimization compared with the baseline method.
- We also show that our method can improve the quality of gradient estimation by the learning rate scheduler function in terms of the *signal-to-noise* ratio (SNR).

The rest of this paper is organized as follows. We overview the background and related studies in Section 2. In Sections 3 and 4, we introduce our framework and discuss its theoretical properties. Finally, we give experimental results and conclusions in Sections 5 and 6.

2. Background

Here, we briefly review VI, the reparameterized gradient, and MCVI in Sections 2.1, 2.2, and 2.3. Furthermore, in Section 2.4, we introduce several related studies of variance reduction in the MCVI context.

2.1 Variational Inference

The aim of Bayesian inference is to estimate the posterior distribution of latent variables $\mathbf{z}$ given the observation $\mathbf{x}$: $p(\mathbf{z}|\mathbf{x})$. The exact computation of $p(\mathbf{z}|\mathbf{x})$ amounts to a sum or integration over all $\mathbf{z}$. However, modern Bayesian statistics rely on a complex (e.g., nonconjugate) model for which the posterior is difficult to compute. Furthermore, the entire inference procedure can be large-scale, and thus performing exact inference is typically intractable. To overcome these problems, we need efficient algorithms for approximating the posterior.

With VI, we can construct an approximation by minimizing the Kullback–Leibler (KL) divergence between the variational distribution $q(\mathbf{z}|\lambda)$ and $p(\mathbf{z}|\mathbf{x})$, where $\lambda \in \mathbb{R}^d$ is a single vector of all free parameters and d is the dimension of the parameter space. This is equivalent to maximizing the evidence lower bound (ELBO) (Jordan et al., 1999; Zhang et al., 2017; Blei et al., 2017):

$$\mathcal{L}(\lambda) = \mathbb{E}_{q(\mathbf{z}|\lambda)}[\log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}|\lambda)], \tag{1}$$

or minimizing the variational free energy (Nakajima et al., 2015):

$$\mathcal{F}(\lambda) = \mathbb{E}_{q(\mathbf{z}|\lambda)}[\log q(\mathbf{z}|\lambda) - \log p(\mathbf{x}, \mathbf{z})]. \tag{2}$$

When VI is applied to large-scale data or a complex model, it is difficult or even sometimes impossible to directly compute the differentiation of the objective in Eqs. 1 and 2 with respect to λ. One way to handle this problem is to use a stochastic gradient on the basis of two major gradient estimators: the score function gradient estimator (Ranganath et al., 2014) and the reparameterized gradient estimator (Titsias and Lázaro-Gredilla, 2014; Kingma and Welling, 2014; Rezende et al., 2014). These gradient estimators are obtained by approximating the expectation of the gradient with independent and identically distributed (i.i.d.) samples from $q(\mathbf{z}|\lambda)$. However, the score function gradient estimator tends to have a higher variance than the reparameterized gradient estimator (Miller et al., 2017; Roeder et al., 2017; Buchholz et al., 2018). That is, the reparameterized gradient estimator has attracted considerable attention as a more convenient tool to reduce the variance than the score function gradient estimator, and has been the subject of extensive research in recent years (Miller et al., 2017; Tucker et al., 2017; Domke, 2019; Xu et al., 2019; Geffner and Domke, 2020).

2.2 Reparameterized Gradient

The use of the reparameterized gradient is a notable approach for learning complex models or reducing the estimation variance based on the reparameterization trick (Kingma and Welling, 2014). In this gradient, the variable $\mathbf{z}$ generated from the distribution $q(\mathbf{z}|\lambda)$ is expressed as a deterministic transformation $\mathcal{T}(\cdot)$ of another simple distribution $p(\epsilon)$ over a noise variable ϵ. Therefore, $\mathbf{z}$ can be expressed as $\mathbf{z} = \mathcal{T}(\epsilon; \lambda)$, where $\epsilon \overset{\text{i.i.d.}}{\sim} p(\epsilon)$. If we assume

that $q(\mathbf{z}|\lambda)$ is in a multivariate location-scale family, we often use an affine transformation $\mathcal{T}(\epsilon;\lambda) = \mathbf{m} + \mathbf{v}\epsilon$ (Titsias and Lázaro-Gredilla, 2014), where $\lambda = (\mathbf{m}, \mathbf{v})$. If we set $p(\epsilon)$ as the standard normal Gaussian $\mathcal{N}(0, I_d)$, for example, $\mathcal{T}(\epsilon;\lambda)$ provides samples from $\mathcal{N}(\mathbf{m}, \mathbf{v}^\top\mathbf{v})$. By using the reparameterization trick, we can express the gradients of the ELBO and the variational free energy as the expectation with respect to $p(\epsilon)$ instead of $q(\mathbf{z}|\lambda)$, that is,

$$\mathbb{E}_{p(\epsilon)}[\nabla_\lambda \log p(\mathbf{x}, \mathcal{T}(\epsilon;\lambda)) - \nabla_\lambda \log q(\mathcal{T}(\epsilon;\lambda)|\lambda))],$$

and

$$\mathbb{E}_{p(\epsilon)}[\nabla_\lambda \log q(\mathcal{T}(\epsilon;\lambda)|\lambda)) - \nabla_\lambda \log p(\mathbf{x}, \mathcal{T}(\epsilon;\lambda))]. \tag{3}$$

In the above, the distribution for calculating the expectation is fixed, and the gradient estimator is obtained by approximating the expectation with i.i.d. random variables ϵ from $p(\epsilon)$. We show that this property is important for the derivation of our method in Section 3.1.

2.3 Monte Carlo Variational Inference (MCVI)

From here, instead of ELBO, we focus on the variational free energy to derive our method based on gradient-descent-based optimization. In the general MCVI framework, the gradient of the variational free energy is represented as an expectation: $\nabla_\lambda \mathcal{F}(\lambda) = \mathbb{E}[g_\lambda(\tilde{\mathbf{z}})]$ over a random variable $\tilde{\mathbf{z}}$, where $g_\lambda(\cdot)$ is a function of the gradient in Eq. 2. For the reparameterization estimator, Eq. 3 with $\tilde{\mathbf{z}} = \epsilon$ leads to the expression $\nabla_\lambda \mathcal{F}(\lambda) = \mathbb{E}_{p(\epsilon)}[g_\lambda(\epsilon)]$, where

$$g_\lambda(\epsilon) = \nabla_\lambda \log q(\mathcal{T}(\epsilon;\lambda)|\lambda) - \nabla_\lambda \log p(\mathbf{x}, \mathcal{T}(\epsilon;\lambda)).$$

To estimate this gradient stochastically, we use an *unbiased* estimator calculated by averaging over i.i.d. samples $\{\epsilon_1, \epsilon_2, \ldots, \epsilon_N\}$, that is,

$$\widehat{\nabla}_{\lambda_t} \mathcal{F}(\lambda_t) = \hat{g}_{\lambda_t}(\epsilon_{1:N}) = \frac{1}{N}\sum_{n=1}^{N} g_{\lambda_t}(\epsilon_n), \tag{4}$$

where t represents the optimization step. The variational free energy can then be optimized on the basis of $\hat{g}_{\lambda_t}(\epsilon_{1:N})$ by using some form of stochastic optimization such as stochastic gradient descent (SGD) updates with a decreasing learning rate $\alpha_t = \alpha_0\eta_t$:

$$\lambda_{t+1} = \lambda_t - \alpha_t \hat{g}_{\lambda_t}(\tilde{\mathbf{z}}). \tag{5}$$

Here, α_0 is the initial learning rate and η_t (> 0) is the value of the learning rate scheduler function, where $\eta_0 = \eta_{-1} = 1$. The learning rate scheduling has been empirically shown to be useful in improving inference performance (Li and Arora, 2020) and is often used to guarantee the convergence of stochastic optimization when the gradient estimator is noisy (Bottou et al., 2016).

2.4 Other Related Studies

Here, we introduce several related studies that have been conducted on variance reduction and SNR for MCVI. Table 1 summarizes the relationships between our study and existing studies.

	Method	OV	GE	SS	CA	SNR
Ranganath et al. (2014)	CV & RB	$\mathcal{O}(N^{-1})$	SF	Stay	-	-
Ruiz et al. (2016a)	IS	$\mathcal{O}(N^{-1})$	SF	Stay	-	-
Roeder et al. (2017)	Stop Gradient	$\mathcal{O}(N^{-1})$	RG	Stay	-	-
Miller et al. (2017)	CV	$\mathcal{O}(N^{-1})$	RG	Stay	-	-
Sakaya and Klami (2017)	IS	$\mathcal{O}(N^{-1})$	SF & RG	Stay	-	-
Li et al. (2018)	Adaptive IS	$\mathcal{O}(N^{-1})$	SF	Stay	-	-
Buchholz et al. (2018)	RQMC	$\mathcal{O}(N^{-2})$	SF & RG	Stay	✓(fixed learning rate)	-
This work	MLMC	$\mathcal{O}(\eta_{t-1}^2 N_t^{-1})$	RG	Adaptive	✓(learning rate scheduling)	✓

Table 1: Relationship between previous work and this work (OV: Order of variance, GE: Gradient estimator, SS: Sample size, CA: Convergence analysis, SNR: SNR analysis). "CV" stands for control variates, "RB" for Rao-Blackwellization, and "IS" for importance sampling. N denotes the number of random variables from $p(\epsilon)$, and t is an iteration step. "SF" means a score function, and "RG" stands for the reparameterized gradient. N_t and η_t denote the adaptively-estimated sample size and the learning rate scheduler function in iteration step t, respectively. "SNR" stands for *signal-to-noise* ratio.

Variance Reduction for MCVI : Since the introduction of MCVI, VI has become a powerful tool for inference on various model architectures. However, MCVI has the crucial problem that the convergence of the stochastic optimization scheme tends to be slow when the magnitude of the gradient estimator's variance becomes high owing to the Monte Carlo estimation. Many techniques have been proposed for variance reduction in this context, such as using control variates (Glasserman, 2003), Rao-Blackwellization (Ranganath et al., 2014), importance sampling (Ruiz et al., 2016a; Burda et al., 2016; Sakaya and Klami, 2017; Li et al., 2018), and many others (Titsias and Lázaro-Gredilla, 2015; Roeder et al., 2017). Nevertheless, it is still challenging to properly construct auxiliary functions such as a control variate function and importance function in these methods.

After the reparameterization trick was proposed by Kingma and Welling (2014) and Rezende et al. (2014), many studies on the reparameterized gradient have also been conducted, e.g., studies on generalized reparameterized gradients (Ruiz et al., 2016b; Tokui and Sato, 2016; Tucker et al., 2017; Grathwohl et al., 2018), control variates on reparameterized gradients (Miller et al., 2017; Geffner and Domke, 2020), and the doubly reparameterized gradient (Tucker et al., 2019). These studies have made the reparameterized gradient estimator a more practical tool fot reducing the gradient variance in the MCVI framework.

The idea of using a more sophisticated method of Monte Carlo sampling to reduce the variance of the estimator has been adopted in a Markov Chain Monte Carlo (MCMC) context, e.g., Lyne et al. (2015) and Georgoulas et al. (2017), but has only recently been explored in the MCVI framework. The objective of this framework is to improve the $\mathcal{O}(N^{-1})$ rate of the gradient estimator's variance. Ranganath et al. (2014) and Ruiz et al. (2016a) suggested using quasi-Monte Carlo (QMC), and Tran et al. (2017) applied it to a specific model. Recently, Buchholz et al. (2018) have proposed a variance reduction method that uses randomized QMC (RQMC), called QMCVI, which can achieve, in the best case, the $\mathcal{O}(N^{-2})$ rate of the variance in the MCVI framework. However, it is known that estimation

with QMC-based methods is sometimes worse than that with MC methods because of a potentially unfavorable interaction between the underlying deterministic points and the function to be estimated (Lemieux, 2009).

In this paper, we focus only on sampling-based methods such as the MC-based method (Ranganath et al., 2014), the RQMC-based method (Buchholz et al., 2018), and our method, and compare their theoretical properties and performance. The reason for this is that these methods are the core methods of gradient estimation and can be combined with the various variance reduction strategies introduced above.

SNR : SNR is a measure of the quality of a stochastic gradient estimator. Recently, Rainforth et al. (2018) have analyzed the behavior of an importance-weighted stochastic gradient in terms of the SNR and revealed differences in the effect of increasing the number of importance weights between inference and generative networks on the gradient estimator in the variational autoencoder (Kingma and Welling, 2014). In addition, there have been several theoretical and empirical analyses of stochastic gradient estimators using the SNR (Hennig, 2013; Lee et al., 2018; Shalev-Shwartz et al., 2017; Tucker et al., 2019).

3. Multilevel Monte Carlo Variational Inference (MLMCVI)

In this section, we introduce our proposed method, MLMCVI. First, we explain the core idea behind proposing MLMCVI in Section 3.1. Finally, we derive its inference algorithm in Section 3.2. A summary of the MLMC method is given in Appendix A.

3.1 Key Idea of MLMCVI

The key idea of MLMCVI is to construct a low-variance gradient estimator by recycling past parameters and gradients that are *information obtained as the optimization proceeds.* The equation in 3 implies that the MLMC method is applicable to the reparameterized gradient since the expectation always depends on a fixed distribution $p(\epsilon)$, and linearity of expectation is available. Another important idea of MLMCVI is to consider the number of levels as the number of iterations t in the optimization process. By applying these ideas, we can "recycle" previous parameters and old gradient estimates. When we set

$$g_{\lambda_t}(\epsilon) = \nabla_{\lambda_t} \log q(\mathcal{T}(\epsilon; \lambda_t)|\lambda_t) - \nabla_{\lambda_t} \log p(\mathbf{x}, \mathcal{T}(\epsilon; \lambda_t)), \tag{6}$$

the Multilevel reparameterized gradient (MLRG) at iteration T is expressed as

$$\nabla_{\lambda_T}^{\mathrm{MLRG}} \mathcal{F}(\lambda_T) = \mathbb{E}_{p(\epsilon)}[g_{\lambda_0}(\epsilon)] + \sum_{t=1}^{T} \left(\mathbb{E}_{p(\epsilon)}[g_{\lambda_t}(\epsilon) - g_{\lambda_{t-1}}(\epsilon)] \right), \tag{7}$$

where $t \in \mathbb{N}$ and $T \in \mathbb{N}$. In the MCVI framework, we need an unbiased estimator of the gradient for stochastic optimization. An unbiased estimator for the MLRG in Eq. 7 can be immediately obtained as

$$\widehat{\nabla}_{\lambda_T}^{\mathrm{MLRG}} \mathcal{F}(\lambda_T) = N_0^{-1} \sum_{n=1}^{N_0} g_{\lambda_0}(\epsilon_{(n,0)}) + \sum_{t=1}^{T} \left(N_t^{-1} \sum_{n=1}^{N_t} [g_{\lambda_t}(\epsilon_{(n,t)}) - g_{\lambda_{t-1}}(\epsilon_{(n,t)})] \right), \tag{8}$$

where N_t $(t = 0, 1, \ldots, T)$ is the sample size in each iteration, and the MLRG estimator is unbiased for $\nabla_{\lambda_T} \mathcal{F}(\lambda_T)$.

We note that the old gradient $g_{\lambda_{t-1}}(\epsilon_{(n,t)})$ is estimated each time using the random variables $\epsilon_{(n,t)}$ per iteration. That is, the past gradient estimated at step t, i.e., $g_{\lambda_{t-1}}(\epsilon_{(n,t)})$, and the gradient estimated at step $t-1$ (which is the target at that time), i.e., $g_{\lambda_{t-1}}(\epsilon_{(n,t-1)})$, are independent and thus uncorrelated.

3.2 Derivation

The MLRG estimator has a problem: the total cost can become crucially large as t goes to infinity because it has a telescoping sum term. To bypass this problem, we consider another formulation of the MLRG estimator and derive a special update rule on the basis of SGD in Eq. 5.

Lemma 1 (Another formulation of MLRG estimator) *The MLRG estimator in iteration $t \geq 1$ can be represented as*

$$\widehat{\nabla}^{\mathrm{MLRG}}_{\lambda_t} \mathcal{F}(\lambda_t) = \widehat{\nabla}^{\mathrm{MLRG}}_{\lambda_{t-1}} \mathcal{F}(\lambda_{t-1}) + N_t^{-1} \sum_{n=1}^{N_t} \Big[g_{\lambda_t}(\epsilon_{(n,t)}) - g_{\lambda_{t-1}}(\epsilon_{(n,t)}) \Big],$$

and the update rule for this estimator in SGD is

$$\lambda_{t+1} = \lambda_t + \frac{\eta_t}{\eta_{t-1}} (\lambda_t - \lambda_{t-1}) - \alpha_t N_t^{-1} \sum_{n=1}^{N_t} \Big[g_{\lambda_t}(\epsilon_{(n,t)}) - g_{\lambda_{t-1}}(\epsilon_{(n,t)}) \Big].$$

Proof *See Appendix B.1 for the proof.* ■

From this result, we can obtain the MLRG estimator by simply saving the previous parameters and computing the two gradients.

How can we estimate the optimal sample size N_t of the MLRG estimator at each t $(t \geq 1)$? In the context of variance reduction, we can consider the optimal sample size N_t as the one that minimizes the *total* variance of the MLRG estimator. In fact, as in the study by Giles (2015), the optimal N_t can be derived by temporarily treating N_t as a real number and solving the constrained optimization problem. To derive this, we set the following definition and assumption.

Definition 1 (One-sample cost and variance) *Let C_t and $\mathbb{V}_t$ be the one-sample computational cost and variance of $g_{\lambda_t}(\epsilon_{(n,t)}) - g_{\lambda_{t-1}}(\epsilon_{(n,t)})$ at iteration t, respectively. At iteration $t = 0$, let C_0 and $\mathbb{V}_0$ be the one-sample computational cost and variance of $g_{\lambda_0}(\epsilon_{(n,0)})$, respectively.*

From the above definition, the total cost and variance of $\widehat{\nabla}^{\mathrm{MLRG}}_{\lambda_T} \mathcal{F}(\lambda_T)$ $(T \geq 1)$ can be expressed as $\sum_{t=0}^{T} N_t C_t$ and $\sum_{t=0}^{T} N_t^{-1} \mathbb{V}_t$, respectively. Since the computational cost of the gradient calculation in each iteration of SGD is $\mathcal{O}(d)$ (Mu et al., 2017), where d is the dimension of the parameter space, i.e., $\lambda \in \mathbb{R}^d$, we can see that $C_0 \leq \nu d$ and $C_t \leq 2\nu d$

($t \geq 1$), where ν is a positive constant. That is, the computational cost of our method does not depend on the iteration t. Throughout this paper, we assume that this upper bound is the actual computational cost as follows.

Assumption 1 (Computational cost of gradient calculation) *The one-sample computational costs of $g_{\lambda_0}(\epsilon_{(n,0)})$ and $g_{\lambda_t}(\epsilon_{(n,t)}) - g_{\lambda_{t-1}}(\epsilon_{(n,t)})$ are $C_0 = \nu d$ and $C_t = 2\nu d$ ($t \geq 1$), respectively.*

Assumption 1 corresponds to a "pessimistic" case for the computational cost of gradient estimation in each iteration. Under Definition 1 and Assumption 1, we can establish the following theorem according to standard proof techniques reported by Giles (2008).

Theorem 1 (Optimal sample size N_t) *Suppose that Assumption 1 holds. Let N_t be a positive real number for all t. Suppose that a total computational cost for sampling is fixed, i.e., $\sum_{t=0}^{T} N_t C_t \leq \sum_{t=0}^{T} C_t N_0$. Then, the optimal sample size for $t \geq 1$, which minimizes the overall variance $\sum_{t=0}^{T} N_t^{-1} \mathbb{V}_t$, is*

$$N_{t+1} = \sqrt{\frac{\mathbb{V}_{t+1}}{\mathbb{V}_t}} N_t, \tag{9}$$

where $N_1 = \frac{(2T+1)\sqrt{\mathbb{V}_1}}{\sqrt{2\mathbb{V}_0} + 2\sum_{t=1}^{T}\sqrt{\mathbb{V}_t}} N_0$.

Proof *(Sketch of Proof) This theorem can be proved by solving a constrained optimization problem that minimizes the overall variance as $\min_{N_t} \sum_{t=0}^{T} N_t^{-1} \mathbb{V}_t$ s.t. $\sum_{t=0}^{T} N_t C_t \leq \sum_{t=0}^{T} C_t N_0$. By solving this for $t \geq 1$, we can obtain $N_t = \frac{1}{\mu}\sqrt{\frac{\mathbb{V}_t}{C_t}}$, where $\mu > 0$. Taking the ratio N_{t+1}/N_t leads to the above result. The complete proof is given in Section B.2.* ■

The constrained optimization problem above implies that we construct the optimal sample size N_t so as to minimize the total variance at a lower cost than when the sample size is fixed as the initial sample size N_0. Furthermore, Theorem 1 shows that the optimal sample size N_{t+1} does not depend on the cost of computing the gradients when N_t is given but depends on the ratio of the *one-sample* variances.

Unfortunately, we still cannot estimate the optimal N_t using Eq. 9 because the true $\mathbb{V}_{t+1}$ and $\mathbb{V}_t$ are unknown. To bypass this problem, we consider an alternative way by minimizing an upper bound on the variance. Therefore, the magnitude of $\mathbb{V}_t$ is critical for sample size estimation. To confirm this perspective, we analyze the *one-sample* gradient variance under the following assumptions, which are often considered in the MCVI context.

Assumption 2 (Variational distribution in a multivariate location-scale family) *The variational distribution $q(\mathcal{T}(\epsilon;\lambda)|\lambda)$ is in a multivariate location-scale family with a single vector of parameters $\lambda = (\mathbf{m}, \mathbf{v})$.*

Assumption 3 (Boundedness of $\mathcal{T}(\epsilon;\lambda)$) *The reparameterized random variable $\mathcal{T}(\epsilon;\lambda)$ is bounded, i.e., $\exists K_1 > 0$ s.t. $\forall\lambda$: $\|\mathcal{T}(\epsilon;\lambda)\|_2^2 \leq K_1$.*

Assumption 4 (Robbins–Monro condition) *The learning rate α_t satisfies the following conditions: $\sum_{t=0}^{\infty} \alpha_t = \infty$ and $\sum_{t=0}^{\infty} \alpha_t^2 < \infty$.*

Assumption 2 has been extensively used in several VI frameworks with a stochastic gradient (Kingma and Welling, 2014; Rezende et al., 2014). In Assumption 3, we also assume that $\mathcal{T}(\epsilon;\lambda)$ is bounded. Although this assumption is not always satisfied, we can make it hold by alternative techniques, e.g., by truncating $\mathcal{T}(\epsilon;\lambda)$ to a particular value by the proximal operator (Nesterov, 1983). The details and the new algorithm obtained from this truncation are described in Appendix C. The Robbins–Monro condition (Assumption 4) is often assumed in convergence analysis on SGD with learning rate scheduling (Bottou et al., 2016).

Under the assumptions above, we give the following proposition for the *one-sample* gradient variance.

Proposition 1 (Order of one-sample gradient variance) *Suppose that Assumptions 2–4 hold. Then, the expectation of the l_2-norm of $g_{\lambda_t}(\epsilon) - g_{\lambda_{t-1}}(\epsilon)$ in iteration t $(t \geq 1)$ is bounded as*

$$\mathbb{E}_{p(\epsilon)}\left[\|g_{\lambda_t}(\epsilon) - g_{\lambda_{t-1}}(\epsilon)\|_2^2\right] \leq \eta_{t-1}^2 K_1(c_1 + d\delta c_2),$$

where δ, c_1, c_2 are positive constants.
Proof *See Appendix B.3 for the proof.* ■

According to Proposition 1, the order of the *one-sample* variance $\mathbb{V}_t$ is $\mathcal{O}(\eta_{t-1}^2)$; therefore, $\mathbb{V}_t \overset{t\to\infty}{\to} 0$. From the above results, we can derive an alternative way for sample size estimation as follows.

Theorem 2 (Alternative sample size estimation) *Suppose that Assumptions 1–4 hold and a total computational cost for sampling is fixed, i.e., $\sum_{t=0}^{T} N_t C_t \leq \sum_{t=0}^{T} C_t N_0$. Let N_t be a positive real number for all t. Then, the optimal sample size for $t \geq 1$, which minimizes the upper bound of the total variance, is*

$$N_{t+1} = \eta_t N_1, \tag{10}$$

where κ is a positive constant and $\frac{2T+1}{2T+\sqrt{2}} N_0 \leq N_1 \leq N_0$.
Proof *See Section B.4 for the proof.* ■

Remark 1 *From Theorems 1 and 2, since the optimal N_t is derived at $t \geq 1$ and N_0 is given as an initial value, we cannot estimate N_1 by using the "optimal" N_0. Therefore, we have to set N_1 so that the following condition holds: $\frac{2T+1}{2T+\sqrt{2}} N_0 \leq N_1 \leq N_0$. From this, we can see $N_1 \simeq N_0$ if we set T to be large enough. When $T = 100$ and $N_0 = 100$, for example, we can see $99.2978 \simeq \frac{201}{200+\sqrt{2}} \cdot 100 \leq N_1 \leq 100$. Therefore, even with the assumption that $N_0 = N_1$, the impact on our algorithm can be viewed as small. From now on, we discuss our method assuming that $N_0 = N_1$.*

Algorithm 1 Multilevel Monte Carlo Variational Inference

Require: Data $\mathbf{x}$, random variable $\epsilon \sim p(\epsilon)$, transform $\mathbf{z} = \mathcal{T}(\epsilon; \lambda)$, model $p(\mathbf{x}, \mathbf{z})$, variational family $q(\mathbf{z}|\lambda)$

Ensure: Variational parameter λ^*

Initialize: N_0, λ_0, α_0, and hyperparameter of η
for $t = 0$ to T **do**
  if $t = 0$ **then**
    $\epsilon_n \sim p(\epsilon)$ $(n = 1, 2, \ldots, N_0)$ $\triangleleft$ sampling ϵ
    $\hat{g}_{\lambda_0}(\epsilon_{1:N_0^*}) = N_0^{-1} \sum_{n=1}^{N_0} g_{\lambda_0}(\epsilon_n)$ $\triangleleft$ calc. RG estimator
    $\lambda_1 = \lambda_0 - \alpha_0 \hat{g}_{\lambda_0}(\epsilon_{1:N_0})$ $\triangleleft$ grad-update
  else
    estimate N_t using $N_t = \lceil \eta_{t-1} N_0 \rceil$
    $\epsilon_n \sim p(\epsilon)$ $(n = 1, 2, \ldots, N_t)$ $\triangleleft$ sampling ϵ
    $\hat{g}'_{\lambda_t}(\epsilon_{1:N_t}) = N_t^{-1} \sum_{n=1}^{N_t} [g_{\lambda_t}(\epsilon_{(n,t)}) - g_{\lambda_{t-1}}(\epsilon_{(n,t)})]$ $\triangleleft$ calc. Multilevel term
    $\lambda_{t+1} = \lambda_t + \frac{\eta_t}{\eta_{t-1}}(\lambda_t - \lambda_{t-1}) - \alpha_t \hat{g}'_{\lambda_t}(\epsilon_{1:N_t})$ $\triangleleft$ grad-update
    if λ_{t+1} has converged to λ^* **then**
      break
    end if
  end if
end for
return λ^*

From Theorem 2, we can estimate the sample size by using only the learning rate scheduler η_t. Since the sample size N_t has to be a positive integer, we use the ceiling function in Eq. 10, i.e., $N_{t+1}^* = \lceil N_{t+1} \rceil = \lceil \eta_t N_1 \rceil$, where $\lceil x \rceil = \min\{k \in \mathbb{Z} | x \leq k\}$. Although the cost constraint condition of Theorem 2 seems to be violated by rounding N_t via the ceiling function, we show that this need not be a concern if we set η_t so that the following definition holds.

Definition 2 (Low total cost condition) *We call the low total cost condition if the total cost added by r_t for each t satisfies the following condition:*

$$\sum_{t=0}^{T} (N_t + r_t) C_t < \sum_{t=0}^{T} C_t N_0,$$

where r_t is a positive real number satisfying $0 < r_t < 1$ and $r_0 = 0$ for all t.

The above condition implies that the total cost added by r_t for all t $(t \geq 1)$ does not exceed the total cost constraint in Theorem 2. Now we show the sufficient condition of the learning scheduler function η_t to satisfy the low total cost condition in Definition 2 as follows.

Lemma 2 (Sufficient conditions for satisfying low total cost condition) *Suppose that Assumption 1 holds. Let N_t be a positive real number for all t and N_{t+1} be estimated as $N_{t+1} = \eta_t N_1$ for $t \geq 1$ where $N_1 = N_0$. If we set η_t to be $\eta_t \leq 1 - \frac{2}{N_0}$ at least after the iteration $\lfloor \frac{T}{2} \rfloor$ $(T \geq 2)$, where $\lfloor x \rfloor = \max\{k \in \mathbb{Z} | k \leq x\}$, the low total cost condition holds.*

Proof *See Appendix B.5 for the proof.* ■

This lemma implies that, if we set the learning rate scheduler function to be less than $1 - \frac{2}{N_0}$ after at least about half of the iteration T, the original constraint condition in Theorem 2 is not violated even if the total cost is increased by $0 < r_t < 1$ for all t. Namely, the optimal N_t is invariant under the above condition even if the original total cost is increased by the ceiling function. Let us consider using the step-based decay function (see Appendix D) and setting $N_0 = 100$, $T = 1000$, and $\{\beta, r\} = \{0.5, 100\}$ as an example. Then, the function η_t needs to be less than $1 - \frac{2}{N_0} = 0.998$ at least in iteration $t = \lfloor \frac{1000}{2} \rfloor = 500$. We can see $\eta_{500} = \beta^{\lfloor \frac{500}{r} \rfloor} = 0.5^{\lfloor \frac{500}{100} \rfloor} = 0.03125 < 0.998$ and the condition holds. To guarantee the result of Lemma 2, we have to assume that $T \geq 2$. However, in MCVI, we often set a sufficiently large T until the optimization converges; therefore, this assumption is quite natural. With this property, if we set the hyperparameter of η_t properly, we can use the ceiling function for the optimal sample size derived in Theorem 2, regardless of the increase in the cost constraint condition. For simplicity, we denote N_t^* as N_t hereafter. Furthermore, since we assume that $N_0 = N_1$, we can rewrite the sample size estimation scheme as $N_t = \lceil \eta_{t-1} N_0 \rceil$, where we denote $\eta_{-1} = \eta_0 = 1$. Straightforwardly, we can see that N_t goes to 1 as $t \to \infty$ because $\eta_{t-1} \to 0$. Therefore, our method can reduce the sample cost for gradient estimation as the optimization proceeds.

According to Lemmas 1–2 and Theorem 2, the MLMCVI algorithm is derived as Algorithm 1.

4. Theoretical Analyses

In this section, we analyze the effect of our method on the basis of the weighted average norm of the gradient and SNR. In addition, we compare the results with sampling-based methods such as MCVI and QMCVI (Buchholz et al., 2018). For this analysis, we add the following assumption, which is often considered in the MCVI context.

Assumption 5 (Lipschitz continuity on $\nabla_\lambda \mathcal{F}(\lambda)$) *The variational free energy $\mathcal{F}(\lambda)$ is a function with Lipschitz continuous derivatives, i.e., $\exists K_2 > 0$ s.t. $\forall \lambda, \bar{\lambda}$: $\|\nabla_\lambda \mathcal{F}(\lambda) - \nabla_{\bar{\lambda}} \mathcal{F}(\bar{\lambda})\|_2^2 \leq K_2 \|\lambda - \bar{\lambda}\|_2^2$.*

This assumption means that the gradient of the variational free energy cannot change very rapidly as λ changes (Buchholz et al., 2018; Domke, 2019).

4.1 Convergence Analysis

Here, we first analyze the convergence of existing methods and our method theoretically and compare them. In our method, since the MLRG estimator appears after the first optimization step, we focus on the case where the number of iterations t is $t \geq 1$. We should first analyze the order of $\mathbb{V}[\widehat{\nabla}_{\lambda_t}^{\mathrm{MLRG}} \mathcal{F}(\lambda_t)]$ so that we can perform the convergence analysis because it has been shown by Buchholz et al. (2018) that the convergence of MCVI depends on the variance of the gradient estimator.

Lemma 3 (Variance of $\widehat{\nabla}_{\lambda_t}^{\mathbf{MLRG}} \mathcal{F}(\lambda_t)$) *Suppose that Assumptions 1–4 hold. Then, the order of $\mathbb{V}[\widehat{\nabla}_{\lambda_t}^{MLRG} \mathcal{F}(\lambda_t)]$ is $\mathcal{O}(\eta_{t-1}^2 N_t^{-1})$.*

Proof *See Section B.6 for the proof.* ■

From the results from Theorem 2 and Lemma 3, and the fact that $\eta_t \to 0$ as $t \to \infty$, we can see that the variance of our method, $\mathbb{V}[\widehat{\nabla}^{\mathrm{MLRG}}_{\lambda_t}\mathcal{F}(\lambda_t)]$, converges to 0.

Regarding stochastic optimization, Bottou et al. (2016) provided comprehensive theorems on the basis of SGD. On the basis of those theorems, we can prove the following upper bounds on the norm of the gradient and use them to analyze the effect of our method. We can also compare the upper bounds with those for the MC-, RQMC-, and MLMC-based methods. Before showing the results, for simplification, we define the common terms in the other expressions as

$$G_T = \frac{1}{A_T}[\mathcal{F}(\lambda_1) - \mathcal{F}(\lambda^*)] + \frac{\alpha_0^2 K_1}{2A_T}\sum_{t=1}^{T}\eta_t^2 \mathbb{E}\left[||\nabla_{\lambda_t}\mathcal{F}(\lambda_t)||_2^2\right],$$

where $A_T = \sum_{t=1}^{T}\alpha_t$, λ^* is the optimal parameter and λ_t is iteratively defined in the SGD update rule.

Theorem 3 (Weighted average norm of gradient) *Suppose that Assumptions 1–5 hold. Then, in each of the MC-, RQMC-, and MLMC-based methods, the weighted average norm of the gradient at iteration $t \geq 1$ is bounded as*

$$\frac{1}{A_T}\sum_{t=1}^{T}\alpha_t \mathbb{E}\left[||\nabla_{\lambda_t}\mathcal{F}(\lambda_t)||_2^2\right] \leq \begin{cases} G_T + \frac{\alpha_0^2 K_1}{2A_T}\sum_{t=1}^{T}\eta_t^2 \frac{\kappa}{N} \ (\mathrm{MC}), \\ G_T + \frac{\alpha_0^2 K_1}{2A_T}\sum_{t=1}^{T}\eta_t^2 \frac{\kappa}{N^2} (\mathrm{RQMC}), \\ G_T + \frac{\alpha_0^2 K_1}{2A_T}\sum_{t=1}^{T}\eta_t^2 \eta_{t-1}^2 \frac{\kappa}{N_t} (\mathrm{MLMC}), \end{cases}$$

respectively.

Proof *See Appendix B.7 and B.8 for the proofs.* ■

Theorem 3 states that the weighted average norm of the squared gradients converges to zero because of $A_T = \sum_{t=1}^{T}\alpha_t = \infty$ and Assumption 4 even if the gradient estimator is noisy. This fact can guarantee that the expectation of the gradient norms of the MC-, RQMC-, and MLMC-based methods asymptotically remains around zero. In addition, the difference in convergence speed between these methods depends on the last term in each of these bounds. While the convergence of the MC- and RQMC-based methods can only be accelerated by increasing the number of samples, i.e., by increasing the gradient estimation cost, our method can be accelerated without increasing the estimation cost (rather, while reducing it since $N_t \to 1$ as $t \to \infty$; see Section 3.2) using the learning rate scheduler function η_{t-1}^2.

4.2 SNR Analysis

SNR is a useful tool to confirm the behavior of gradient estimation. Recently, for example, Rainforth et al. (2018) have been analyzed using SNR the gradient behavior of various importance-weighted autoencoder. In the context of variance reduction, it is also important to investigate how our method affects gradient estimation and how it changes compared with existing methods to theoretically guarantee its performance. Therefore, we evaluate the quality of MC-, RQMC- and MLMC-based gradient estimators through the SNR analysis.

The SNR of a gradient estimator is defined as

$$\mathrm{SNR}(\lambda) = \frac{\|\mathbb{E}_{p(\epsilon_{1:N})}[\hat{g}_\lambda(\epsilon_{1:N})]\|_2^2}{\sqrt{\mathbb{V}[\hat{g}_\lambda(\epsilon_{1:N})]}},$$

where $\hat{g}_\lambda(\epsilon_{1:N})$ is given by Eq. 4. This indicates that if $\mathrm{SNR} \to 0$, the gradient estimator is dominated by random noise, which degrades the accuracy of estimation.

On the basis of the theoretical properties from Lemma 3, we prove the following theorem for SNR.

Theorem 4 (Signal-to-noise ratio bound) *Suppose that Assumptions 1–5 hold. Then, the SNR at iteration t for each method is bounded as*

$$\mathrm{SNR}(\lambda_t) \geq \begin{cases} \frac{\|\nabla_{\lambda_t}\mathcal{F}(\lambda_t)\|_2^2}{\sqrt{\kappa}} \cdot \sqrt{N} & \text{(MC)}, \\ \frac{\|\nabla_{\lambda_t}\mathcal{F}(\lambda_t)\|_2^2}{\sqrt{\kappa}} \cdot N & \text{(RQMC)}, \\ \frac{\|\nabla_{\lambda_t}\mathcal{F}(\lambda_t)\|_2^2}{\sqrt{\kappa}} \cdot \frac{\sqrt{N_t}}{\eta_{t-1}} & \text{(MLMC)}, \end{cases}$$

where κ is a positive constant and N is an initial sample size for the MC- and RQMC-based methods.

Proof *See Section B.9 for the proof.* ■

The above theorem implies that for the MC- and RQMC-based methods, the SNR can be increased only by increasing the initial sample size, i.e., by increasing the estimation cost. Furthermore, the SNRs of these methods gradually decrease because the sample size N is fixed and $||\nabla_{\lambda_t}\mathcal{F}(\lambda_t)||_2^2$ approaches 0 as the optimization proceeds. This fact means that the gradient estimator based on these methods becomes dominated by random noise. On the other hand, in our method, it seems that the SNR gradually becomes lower than those of the MC- and RQMC-based methods since the sample size N_t decreases as shown in Section 3.2. However, it can be seen that our method can keep it at a high value since the learning rate scheduler function η_{t-1} appears in the denominator. That is, our method can improve the quality of gradient estimation by the learning rate scheduler function, i.e., not by increasing the estimation cost.

5. Experiments

In this section, we carried out experiments to analyze the optimization and prediction performance of our method using three models, which have become benchmark experiments in the context of variance reduction: hierarchical linear regression (HLR), Bayesian logistic regression (BLR), and Bayesian neural network (BNN) regression (Miller et al., 2017; Buchholz et al., 2018), and we compared the results with those of existing methods. First, we compared the optimization performance using ELBO and log-likelihood for the training and test datasets. Note that the reason for using ELBO in the experimental results is for consistency with the experimental setup of related studies (e.g., Miller et al. (2017) and Buchholz et al. (2018)), and that ELBO and variational free energy are not essentially different. Second,

we compared the performance of variance reduction by measuring the empirical gradient variance. Finally, we determined the quality of the gradient estimator on the basis of the empirical SNR. Throughout all experiments, we used Algorithm 1 for our method.

5.1 Experiments on Benchmark Dataset

Here, we report the results of experiments conducted to validate the effectiveness of our method.

5.1.1 Model setting and benchmark dataset

In our experiments, we used three different settings: HLR, BLR, and BNN. A brief summary of the model settings and benchmark datasets used in the experiment is as follows.

Hierarchical Linear Regression (HLR): We applied HLR to toy data generated from the same generation process of model setting. Then, the dimension of the entire parameter space was $d = 1012$.

Bayesian Logistic Regression (BLR): We applied BLR to the breast cancer dataset in the UCI Machine Learning Repository[1]. Then, the dimension of the entire parameter space was $d = 62$.

Bayesian Neural Network (BNN): We applied BNN regression to the wine-quality-red dataset, which is included in the wine-quality dataset in the UCI Machine Learning Repository[2]. Then, the dimension of the entire parameter space is $d = 653$.

We approximated these models using a variational diagonal Gaussian distribution. A more detailed description of the settings for each model is in Appendices F.1, F.2, and F.3.

5.1.2 Experimental Settings

Two baseline methods were used in the experiments: vanilla MCVI based on MC sampling and QMCVI based on RQMC sampling (Buchholz et al., 2018). As described by Buchholz et al. (2018), the RQMC samples were generated using the R package randtoolbox[3].

In the experiments, the parameters λ were initialized to be of the same values for each method. For the optimization of variational free energy, we used Adam for the MC- and RQMC-based methods and the SGD optimizer with the learning rate scheduler η for our method. Furthermore, we adopted a step-based decay function as the learning rate scheduler. The initial learning rate and the hyperparameters of the learning rate scheduler (i.e., β and r) were optimized through the Tree-structured Parzen Estimator (TPE) sampler in optuna[4] (Akiba et al., 2019) in 50 trials, where β and r are the decay and drop-rate parameters, respectively. The selected parameters are summarized in Appendix G (Table 3). In all experiments, the training ELBO and the test log-likelihood were estimated with 2000 MC samples. The gradient variance was estimated by resampling the MC sample and

1. https://archive.ics.uci.edu/ml/datasets/Breast+Cancer
2. https://archive.ics.uci.edu/ml/datasets/Wine+Quality
3. https://cran.r-project.org/web/packages/randtoolbox/index.html
4. optuna is a hyperparameter optimization framework introduced by Akiba et al. (2019) from Preferred Networks, Inc. Thanks to *define-by-run API*, we are allowed to construct the parameter search space dynamically and flexibly. https://optuna.org/

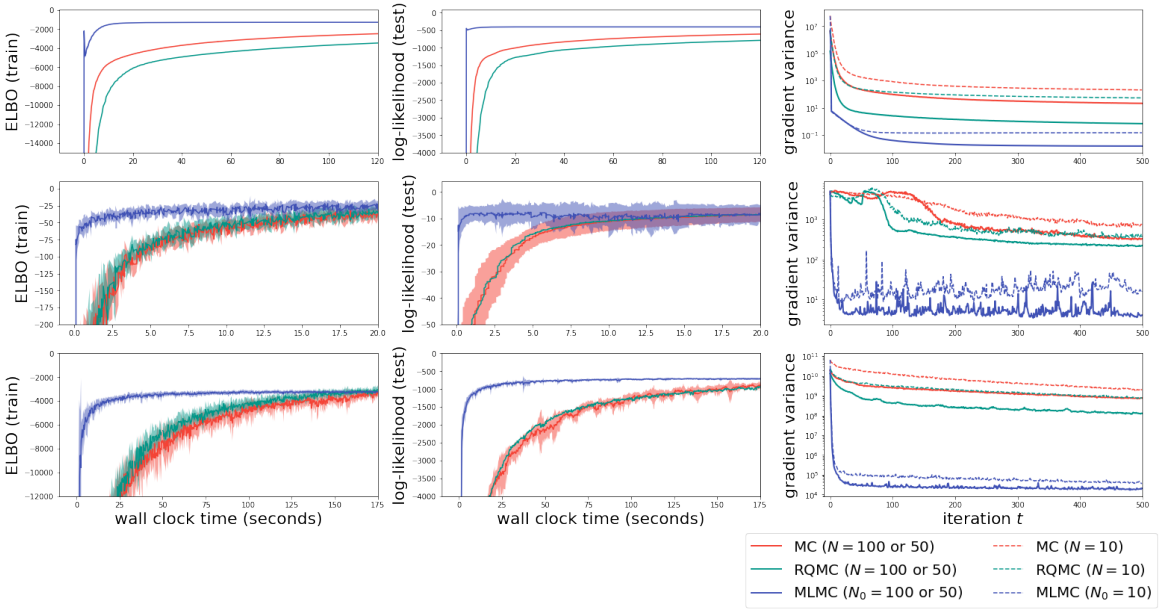

Figure 2: Experimental results. From left to right, the graphs show training ELBO (higher is better; mean ± std), test log-likelihood (higher is better; mean ± std), and empirical variance (lower is better). From top to bottom, results for HLR, BLR, and BNN.

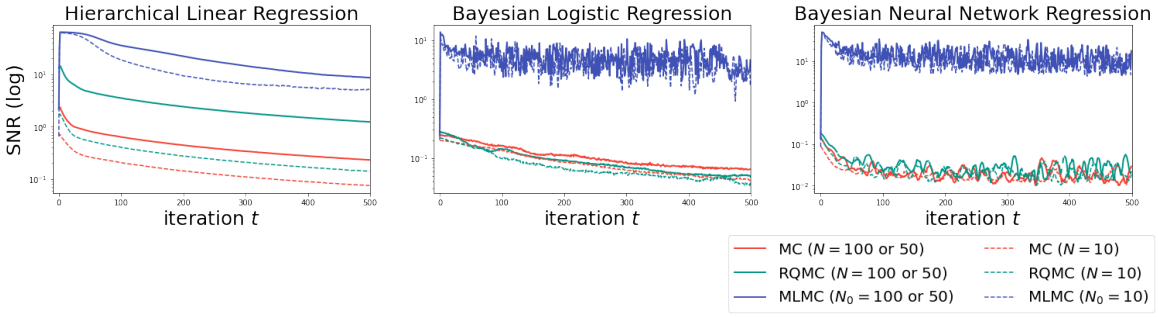

Figure 3: Experimental results on empirical SNR (higher is better).

calculating the variance of the gradient 1000 times and computing the empirical variance at each optimization step. In each experiment, the dataset was randomly divided into training data and test data at the ratio of 8:2. Each experiment was repeated 10 times.

5.1.3 Results

The main experimental results are shown in Figure 2. These results show that our method (solid blue line) converged faster than the baseline methods did in all settings. Furthermore,

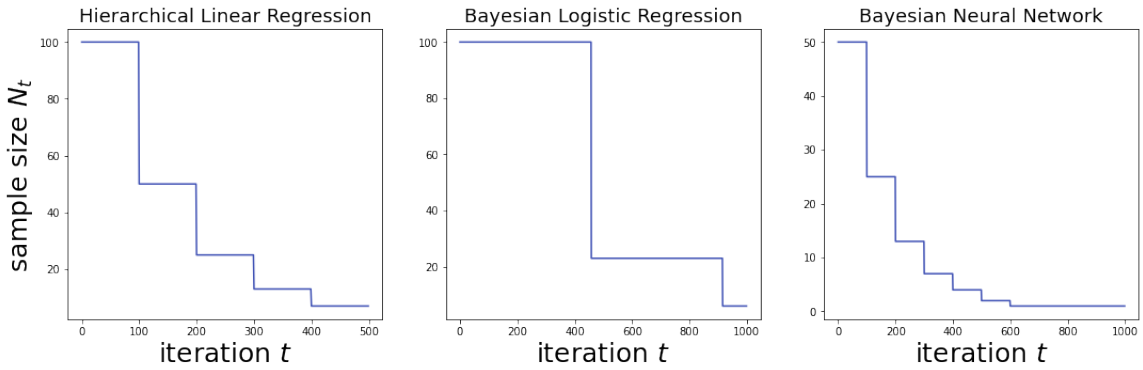

Figure 4: Behavior of estimated sample size.

the results of gradient variance (rightmost column in Figure 2) confirm that our method showed an empirically lower variance gradient estimate than the baseline methods.

The results on empirical SNR are shown in Figure 3. These results show that the empirical SNR of our method (dashed line and solid blue line) is higher than those of the baseline methods, corresponding to the empirical variance. This implies that our method provides better gradient estimation than the baseline method in terms of SNR.

Furthermore, we show the behavior of the estimated sample size N_t in our method in Figure 4. The results show that our method achieves faster convergence while reducing the sample size for gradient estimation.

As we mentioned regarding several theoretical analyses in Section 4, the optimization performance of our method depends on the learning rate scheduler function η. To understand the characteristics of our method in more detail, we conducted benchmark experiments with various hyperparameter settings and initial learning rates to determine how the performance of the optimization changes and compare it with those with the optimal hyperparameter settings. The results are shown in the Appendix G. From these results, we confirmed that, in practice, our method requires the careful optimization of the hyperparameters, especially the drop-rate r, and the initial learning rate.

5.2 Bayesian Logistic Regression for Multiclass Classification

To confirm the performance using a more realistic problem setting and dataset, we conducted experiments on a multiclass classification task for a large image dataset such as the Fashion-MNIST dataset.

5.2.1 Experimental settings and Results

We used Bayesian logistic regression in this experiment. The model settings are described in Appendix F.2. In these settings, the dimension of the entire parameter space is $d = 7852$, and this model was also approximated by a variational diagonal Gaussian distribution. To analyze only the effect of variance reduction and to fairly compare our method with the baseline methods, the variational free energy was optimized using the SGD optimizer with

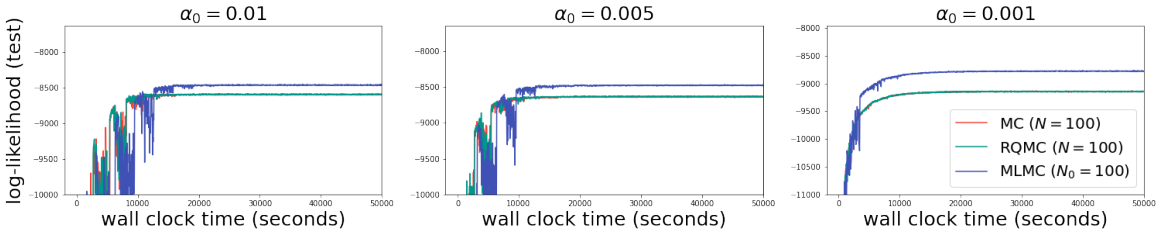

Figure 5: Test log-likelihood (higher is better) for Bayesian logistic regression on the Fashion-MNIST dataset when the initial learning rate $\alpha_0 = \{0.01, 0.005, 0.001\}$.

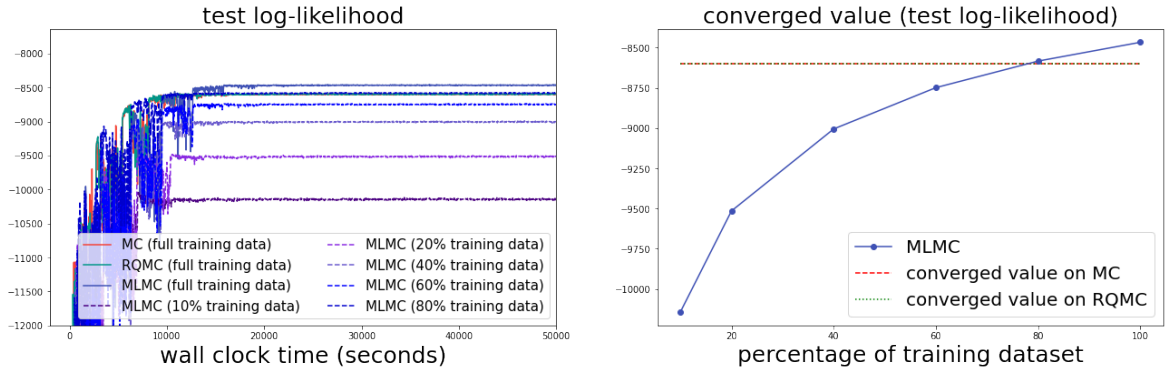

Figure 6: Experimental results when the size of training data changes. Test log-likelihood (higher is better) and its converged value (higher is better) for each percentage are lined up from the left.

the learning rate scheduler function η for all methods. We used 100 initial MC or RQMC samples for gradient estimation. In the optimization step, we used η as the step-decay function and set the hyperparameter $\{\beta, r\}$ for sample size estimation to $\{0.5, 100\}$. Finally, we set the initial learning rate as 0.01, 0.005, or 0.001.

From the results shown in Figure 5, we confirmed that our method provides a better log-likelihood than existing methods, although the convergence speed is almost comparable.

5.2.2 Experimental Results on Data-size Change

We have confirmed that our method achieves a better test log-likelihood than the baseline method from the experimental results above. Then, how many training datasets are sufficient for our method to achieve better performance than the baseline method? To answer this question, we performed the following experiments.

Firstly, we adopted Bayesian logistic regression in this experiment and used the same settings as those in the experiments described in Section 5.2.1. Next, we generated five

training datasets with 10%, 20%, 40%, 60%, and 80% of the original training data. Then, we performed inference on the five reduced training datasets and calculated the log-likelihood for the test data.

The results are shown in Figure 6. From this result, we can see that, even though the training datasets were reduced by about 20%, our method achieved almost the same performance as the baseline method performed on the complete training data. In other words, our method has the potential to perform inference more effectively with fewer data than existing methods.

6. Conclusion

We have proposed a new framework for MCVI with the reparameterized gradient estimator, Multilevel Monte Carlo variational inference (MLMCVI). In the MLMCVI framework, a unique parameter update scheme is naturally derived from SGD. Moreover, the sample size for gradient estimation can be adaptively determined according to the gradient variance, which provides the minimum total variance at each optimization step. Our method can be easily implemented in an automated inference library with an automatic differentiation toolbox, since our method only requires storing the past parameters and using them to estimate the old gradients.

From a theoretical perspective, we have shown that the convergence of the weighted average gradient norm is accelerated and that the learning rate scheduler function can help reduce the degradation of the quality of the MLRG estimator. Furthermore, three benchmark experiments confirmed that our method achieved comparable or better performance in terms of convergence speed, variance reduction, and SNR than the baseline methods. We also confirmed that our method provides a higher test log-likelihood in some cases than the baseline methods.

Since our method uses pure MC sampling, it can be combined with previously proposed variance reduction methods for the reparameterized gradient based on control variates and importance sampling. It is also possible to use RQMC sampling by extending the methods in previous studies, such as those by Giles and Waterhouse (2009) and Gerstner and Noll (2013), to our method.

There are two limitations of our method. First, the theoretical guarantee is shown only when the Gaussian distribution is set as the variational distribution, although commonly used in MCVI; thus, the performance is not guaranteed when other distributions are adopted. Second, in some cases, a truncation is required for $\mathcal{T}(\epsilon; \lambda)$ from Assumption 3, which may cause biases in the gradient estimation. To overcome these limitations, it is necessary to theoretically guarantee the performance of our method under more general variational distributions and analyze the bias induced by truncation of $\mathcal{T}(\epsilon; \lambda)$. We leave this part as our future work.

Our idea of recycling past gradients may be useful not only in MCVI but also in the context of stochastic optimization. Therefore, as our future research, we plan to extend our method to a general stochastic optimization algorithm.

Acknowledgments

We thank Dr. Ikko Yamane, Dr. Futoshi Futami, and Soma Yokoi for their helpful discussions and comments. We also thank Dr. Ikko Yamane, Kento Nozawa, Dr. Yoshihiro Nagano, and Han Bao for maintaining the experimental environment. We also thank Dr. Naonori Ueda, Dr. Tomoharu Iwata, Naoki Marumo, Dr. Masakazu Ishihata, and Dr. Takuma Otsuka of NTT Communication Science Laboratory for their insightful discussions. We would like to express our deepest gratitude to Professor Masashi Sugiyama for his fruitful advice on our paper. We are also grateful to many anonymous reviewers who provided insightful reviews at various international conferences over nearly three and a half years before this paper was accepted for publication in JMLR. We would like to thank the reviewers of JMLR and Dr. Erik Sudderth for their consideration of this paper. This work was supported by RIKEN Junior Research Associate Program until March, 2021. M.F. was supported by the University of Tokyo Toyota-Dwango Scholarship for Advanced AI Talents. M.F. was also supported by JSPS KAKENHI Grant Number 21J11859.

Appendix A. Additional Information on MLMC Method

We introduce additional information on the MLMC method for ease in understanding the background, method, algorithm, and theoretical analysis of our method.

A.1 Sampling Method and Multilevel Monte Carlo (MLMC)

When we approximate posterior distributions, Monte Carlo methods are often used for estimating the expectation of intractable objects with several random samples. The mean squared error (MSE) of approximation with random samples is a rate of $\mathcal{O}(N^{-1})$, and an accuracy of ϵ requires $N = \mathcal{O}(\epsilon^{-2})$ samples. This rate can be too high for application. One approach to addressing this high cost is the use of the QMC or RQMC method, in which the samples are not chosen randomly and independently; instead, they are selected very carefully to reduce the error (Giles, 2015). In the best cases, the error rate is $\mathcal{O}(N^{-2}\log N^{2d-2})$. There are many reviews about the QMC approach provided by Niederreiter (1992), L'Ecuyer and Lemieux (2005), and Leobacher and Pillichshammer (2014).

Another approach to improving the computational efficiency is the MLMC method proposed by Heinrich (2001). This method has been often used in stochastic differential equations for options pricing (Giles, 2008; Cliffe et al., 2011; Rhee and Glynn, 2015). In statistics, there are many applications in approximate Bayesian computation (Giles et al., 2016; Jasra et al., 2017; Warne et al., 2018). In a Bayesian framework, Giles et al. (2016, 2020) applied MLMC to stochastic gradient MCMC algorithms such as the stochastic gradient Langevin dynamics (SGLD), which discretize the posterior of SDE based on the Multilevel step size and couple them.

Because of the linearity of expectations, given a sequence $P_0, P_1, \ldots, P_{L-1}$ that approximates P_L with increasing accuracy, we have a simple identity:

$$\mathbb{E}[P_L] = \mathbb{E}[P_0] + \sum_{l=1}^{L} \mathbb{E}[P_l - P_{l-1}]. \tag{11}$$

We can thus use the following unbiased estimator for $\mathbb{E}[P_L]$,

$$\mathbb{E}[P_L] \approx N_0^{-1} \sum_{n=1}^{N_0} P_0^{(0,n)} + \sum_{l=1}^{L} \left\{ N_l^{-1} \sum_{n=1}^{N_l} (P_l^{(l,n)} - P_{l-1}^{(l,n)}) \right\}, \tag{12}$$

with the inclusion of l in (l, n), indicating that independent samples are used at each level of correction.

When we define V_0, C_0 to be the variance and cost of *one sample* of P_0, and $\mathbb{V}_l, C_l$ to be the variance and cost of *one sample* of $P_l - P_{l-1}$, then, the total variance and cost in Eq. 12 are $\sum_{l=0}^{L} N_l^{-1} \mathbb{V}_l$ and $\sum_{l=0}^{L} N_l C_l$, respectively.

Thus, if Y is a Multilevel estimator given by

$$Y = \sum_{l=0}^{L} Y_l, \; Y_l = N_l^{-1} \sum_{n=1}^{N_l} (P_l^{(l,n)} - P_{l-1}^{(l,n)}),$$

with $P_{-1} \equiv 0$, then

$$\mathbb{E}[Y] = \mathbb{E}[P_L], \; V[Y] = \sum_{l=0}^{L} N_l^{-1} \mathbb{V}_l, \; \mathbb{V}_l \equiv V[P_l - P_{l-1}].$$

The MLMC method has been described in detail by Heinrich (2001) and Giles (2008, 2015).

A.2 Control Variates and Their Relationship with Two-level MLMC

One of the classic methods to reduce the variance of Monte Carlo samples is control variates (Glasserman, 2003). Let us consider that we want to estimate the expectation of a function f, i.e., $\mathbb{E}[f]$. Then, in control variates, we construct a function h that is highly correlated with f with a known expectation $\mathbb{E}[h]$ and estimate $\mathbb{E}[f]$ via its unbiased estimator from N i.i.d. samples $\omega^{(n)}$ as follows:

$$N^{-1} \sum_{n=1}^{N} \{ f(\omega^{(n)}) - a(h(\omega^{(n)} - \mathbb{E}[h])) \}. \tag{13}$$

Then, variance is expressed as

$$V[f(\omega^{(n)})] = V[f(\omega^{(n)})] + a^2 V[h(\omega^{(n)})] - 2a\mathrm{Cov}(f(\omega^{(n)}), h(\omega^{(n)})),$$

and the optimal value for a is $\rho\sqrt{V[f]/V[h]}$, where ρ is the correlation between f and h. Thus, the variance of this estimator is reduced by a factor of $1 - \rho^2$ (Giles, 2008).

The two-level MLMC method is similar to this method. According to Giles (2013), if we want to estimate $\mathbb{E}[P_1]$ but it is much cheaper to simulate P_0 that approximates P_1, then since

$$\mathbb{E}[P_1] = \mathbb{E}[P_0] + \mathbb{E}[P_1 - P_0], \tag{14}$$

we can use the unbiased two-level estimator

$$N_0^{-1} \sum_{n=1}^{N_0} P_0^{(n)} + N_1^{-1} \sum_{n=1}^{N_1} \left(P_1^{(n)} - P_0^{(n)} \right). \tag{15}$$

There are two different points regarding control variates: the value of $\mathbb{E}[P_0]$ is *unknown* and a takes one.

Appendix B. Proofs

Here, we show the proofs of lemmas and theorems introduced in this paper.

B.1 Proof of Lemma 1

Proof MLRG at iteration T can be rewritten as

$$\begin{aligned}\nabla^{\mathrm{MLRG}}_{\lambda_T}\mathcal{F}(\lambda_T) &= \mathbb{E}_{p(\epsilon)}[g_{\lambda_0}(\epsilon)] + \sum_{t=1}^{T-1}\left(\mathbb{E}_{p(\epsilon)}\left[g_{\lambda_t}(\epsilon) - g_{\lambda_{t-1}}(\epsilon)\right]\right) + \mathbb{E}_{p(\epsilon)}\left[g_{\lambda_T}(\epsilon) - g_{\lambda_{T-1}}(\epsilon)\right] \\ &= \nabla^{\mathrm{MLRG}}_{\lambda_{T-1}}\mathcal{F}(\lambda_{T-1}) + \mathbb{E}_{p(\epsilon)}\left[g_{\lambda_T}(\epsilon) - g_{\lambda_{T-1}}(\epsilon)\right].\end{aligned}$$

By constructing the unbiased estimator in the above $\nabla^{\mathrm{MLRG}}_{\lambda_T}\mathcal{F}(\lambda_T)$, we obtain

$$\widehat{\nabla}^{\mathrm{MLRG}}_{\lambda_T}\mathcal{F}(\lambda_T) = \widehat{\nabla}^{\mathrm{MLRG}}_{\lambda_{T-1}}\mathcal{F}(\lambda_{T-1}) + N_T^{-1}\sum_{n=1}^{N_T}\left[g_{\lambda_T}(\epsilon_{(n,T)}) - g_{\lambda_{T-1}}(\epsilon_{(n,T)})\right].$$

From the above result, we have the following update rule according to the SGD:

$$\begin{aligned}\lambda_{T+1} &= \lambda_T - \alpha_T\left(\widehat{\nabla}^{\mathrm{MLRG}}_{\lambda_{T-1}}\mathcal{F}(\lambda_{T-1}) + N_T^{-1}\sum_{n=1}^{N_T}\left[g_{\lambda_T}(\epsilon_{(n,T)}) - g_{\lambda_{T-1}}(\epsilon_{(n,T)})\right]\right) \\ &= \lambda_T - \frac{\eta_T}{\eta_{T-1}}\alpha_0\eta_{T-1}\widehat{\nabla}^{\mathrm{MLRG}}_{\lambda_{T-1}}\mathcal{F}(\lambda_{T-1}) - \alpha_T N_T^{-1}\sum_{n=1}^{N_T}\left[g_{\lambda_T}(\epsilon_{(n,T)}) - g_{\lambda_{T-1}}(\epsilon_{(n,T)})\right] \\ &= \lambda_T + \frac{\eta_T}{\eta_{T-1}}(\lambda_T - \lambda_{T-1}) - \alpha_T N_T^{-1}\sum_{n=1}^{N_T}\left[g_{\lambda_T}(\epsilon_{(n,T)}) - g_{\lambda_{T-1}}(\epsilon_{(n,T)})\right].\end{aligned}$$

The claim is proved by changing T to t. ■

B.2 Proof of Theorem 1

Proof According to Assumption 1, the constrained minimization problem we consider can be expressed as

$$\min_{N_t}\sum_{t=0}^{T} N_t^{-1}\mathbb{V}_t \quad \text{s.t.} \quad \sum_{t=0}^{T} N_t C_t \leq M,$$

where $M = \sum_{t=0}^{T} C_t N_0$. For a fixed cost M, the variance is minimized by choosing N_t to minimize

$$f(N_t) = \sum_{t=0}^{T} N_t^{-1}\mathbb{V}_t + \mu^2\left(\sum_{t=0}^{T} N_t C_t - M\right), \tag{16}$$

for some value of the Lagrange multiplier μ^2 ($\mu > 0$). By solving the above with respect to N_t, we obtain

$$N_t = \frac{1}{\mu}\sqrt{\frac{\mathbb{V}_t}{C_t}} \ (\because \mu > 0, C_t > 0, \mathbb{V}_t > 0). \tag{17}$$

By substituting 17 for 16, we obtain

$$f(N_t) = 2\mu \sum_{t=0}^{T} \sqrt{\mathbb{V}_t C_t} - \mu^2 M. \tag{18}$$

By differentiating $f(N_t)$ in 18 and solving it with respect to μ, we obtain

$$\mu = \frac{1}{M} \sum_{t=0}^{T} \sqrt{\mathbb{V}_t C_t}. \tag{19}$$

We substitute (19) for (17):

$$N_t = \frac{M}{\sum_{t=0}^{T} \sqrt{\mathbb{V}_t C_t}} \sqrt{\frac{\mathbb{V}_t}{C_t}}. \tag{20}$$

By considering the ratio of N_{t+1} to N_t for $t \geq 1$ and according to Assumption 1, we obtain

$$\begin{aligned}
\frac{N_{t+1}}{N_t} &= \frac{M}{\sum_{t=0}^{T} \sqrt{\mathbb{V}_t C_t}} \sqrt{\frac{\mathbb{V}_{t+1}}{C_{t+1}}} \cdot \frac{\sum_{t=0}^{T} \sqrt{\mathbb{V}_t C_t}}{M} \sqrt{\frac{C_t}{\mathbb{V}_t}} \\
&= \sqrt{\frac{\mathbb{V}_{t+1}}{2\nu d}} \cdot \sqrt{\frac{2\nu d}{\mathbb{V}_t}} \\
&= \sqrt{\frac{\mathbb{V}_{t+1}}{\mathbb{V}_t}}.
\end{aligned}$$

According to this result, the optimal sample size N_t can be expressed as

$$N_{t+1} = \sqrt{\frac{\mathbb{V}_{t+1}}{\mathbb{V}_t}} N_t,$$

for $t = 1, \ldots, T$. Furthermore, from Eq. 20, we can see that when $t = 1$,

$$\begin{aligned}
N_1 &= \frac{M}{\sum_{t=0}^{T} \sqrt{\mathbb{V}_t C_t}} \sqrt{\frac{\mathbb{V}_1}{C_1}} \\
&= \frac{\nu d N_0 + 2\nu d N_0 T}{\sqrt{\nu d \mathbb{V}_0} + \sqrt{2\nu d} \sum_{t=1}^{T} \sqrt{\mathbb{V}_t}} \sqrt{\frac{\mathbb{V}_1}{2\nu d}} \\
&= \frac{(2T+1)\sqrt{\mathbb{V}_1}}{\sqrt{2\mathbb{V}_0} + 2\sum_{t=1}^{T} \sqrt{\mathbb{V}_t}} N_0.
\end{aligned}$$

Thus, the claim is proved. ■

B.3 Proof of Proposition 1

Proof According to Assumption 3, we obtain the Lipschitz condition given by

$$||g_{\lambda_t}(\epsilon) - g_{\lambda_{t-1}}(\epsilon)||_2^2 \leq K_1||\mathcal{T}(\epsilon;\lambda_t) - \mathcal{T}(\epsilon;\lambda_{t-1})||_2^2 < \infty.$$

According to Assumption 2, all of the variational parameters can be expressed as a single vector $\lambda = (\mathbf{m}, \mathbf{v})$ and its transformation $\mathcal{T}(\epsilon;\lambda_t)$ can be written as $\mathcal{T}(\epsilon;\lambda_t) = \mathbf{m}_t + \mathbf{v}_t \cdot \epsilon$. From these assumptions, we obtain

$$\begin{aligned}
||g_{\lambda_t}(\epsilon) - g_{\lambda_{t-1}}(\epsilon)||_2^2 &\leq K_1||\mathcal{T}(\epsilon;\lambda_t) - \mathcal{T}(\epsilon;\lambda_{t-1})||_2^2 \\
&= K_1\|(\mathbf{m}_t - \mathbf{m}_{t-1}) + (\mathbf{v}_t - \mathbf{v}_{t-1}) \cdot \epsilon\|_2^2 \\
&\leq K_1(\|\mathbf{m}_t - \mathbf{m}_{t-1}\|_2^2 + \|(\mathbf{v}_t - \mathbf{v}_{t-1}) \cdot \epsilon\|_2^2) \\
&\leq K_1(\|\mathbf{m}_t - \mathbf{m}_{t-1}\|_2^2 + \|\mathbf{v}_t - \mathbf{v}_{t-1}\|_2^2 \cdot \|\epsilon\|_2^2). \qquad (21)
\end{aligned}$$

The third and fourth lines are obtained using the triangle inequality and Cauchy–Schwarz inequality, respectively. Because of the MLRG update rule with decreasing learning rate from Lemma 1, we obtain

$$\begin{aligned}
\|\mathbf{m}_t - \mathbf{m}_{t-1}\|_2^2 &= \left\| \frac{\eta_{t-1}}{\eta_{t-2}}(\mathbf{m}_{t-1} - \mathbf{m}_{t-2}) - \alpha_{t-1}A_{t-1} \right\|_2^2 \\
&= \left\| \frac{\eta_{t-1}}{\underbrace{\eta_0}_{=1}}(\mathbf{m}_1 - \mathbf{m}_0) - \frac{\eta_{t-1}}{\underbrace{\eta_0}_{=1}} \cdot \alpha_1 A_1 - \frac{\eta_{t-1}}{\eta_1} \cdot \alpha_2 A_2 - \cdots - \frac{\eta_{t-1}}{\eta_{t-3}} \cdot \alpha_{t-2}A_{t-2} - \alpha_{t-1}A_{t-1} \right\|_2^2 \\
&= \eta_{t-1}^2 \left\| - \alpha_0 \hat{g}_{\mathbf{m}_0}(\epsilon_{1:N_0}) - \alpha_1 A_1 - \alpha_0 \cdot \frac{\eta_2}{\eta_1} A_2 - \cdots - \alpha_0 \cdot \frac{\eta_{t-2}}{\eta_{t-3}} A_{t-2} - \alpha_0 A_{t-1} \right\|_2^2 \\
&= \alpha_0^2 \eta_{t-1}^2 \left\| \hat{g}_{\mathbf{m}_0}(\epsilon_{1:N_0}) + \eta_1 A_1 + \frac{\eta_2}{\eta_1} A_2 + \cdots + \frac{\eta_{t-2}}{\eta_{t-3}} A_{t-2} + A_{t-1} \right\|_2^2.
\end{aligned}$$

At iteration t, the term $S = \hat{g}_{\mathbf{m}_0}(\epsilon_{1:N_0}) + \eta_1 A_1 + \frac{\eta_2}{\eta_1} A_2 + \cdots + \frac{\eta_{t-2}}{\eta_{t-3}} A_{t-2}$ is constant. By using the triangle inequality, we obtain

$$\|\mathbf{m}_t - \mathbf{m}_{t-1}\|_2^2 = \alpha_0^2 \eta_{t-1}^2 \|S + A_{t-1}\|_2^2 \leq \alpha_0^2 \eta_{t-1}^2 (\|S\|_2^2 + \|A_{t-1}\|_2^2).$$

By taking the expectation over $p(\epsilon)$, we obtain

$$\mathbb{E}_{p(\epsilon)}[\|\mathbf{m}_t - \mathbf{m}_{t-1}\|_2^2] \leq \alpha_0^2 \eta_{t-1}^2 (\|S\|_2^2 + \mathbb{E}_{p(\epsilon)}[\|A_{t-1}\|_2^2]).$$

Here,

$$\begin{aligned}
\mathbb{E}_{p(\epsilon)}[\|A_{t-1}\|_2^2] &= \mathbb{E}_{p(\epsilon)} \left[\left\| N_{t-1}^{-1} \sum_{n=1}^{N_{t-1}} \left[g_{\lambda_{t-1}}(\epsilon_{(n,t-1)}) - g_{\lambda_{t-2}}(\epsilon_{(n,t-1)}) \right] \right\|_2^2 \right] \\
&\leq \mathbb{E}_{p(\epsilon)} \left[\|g_{\lambda_{t-1}}(\epsilon_{(1,t-1)}) - g_{\lambda_{t-2}}(\epsilon_{(1,t-1)})\|_2^2 \right] \\
&\leq K_1 \mathbb{E}_{p(\epsilon)} \left[\|\mathcal{T}(\epsilon_{(1,t-1)};\lambda_{t-1}) - \mathcal{T}(\epsilon_{(1,t-1)};\lambda_{t-2})\|_2^2 \right] < \infty \text{ (Assumption 3)},
\end{aligned}$$

where we assume that $\max\{g_{\lambda_{t-1}}(\epsilon_{(n,t-1)}) - g_{\lambda_{t-2}}(\epsilon_{(n,t-1)}) | 1 \leq n \leq N_{t-1}\} = g_{\lambda_{t-1}}(\epsilon_{(1,t-1)}) - g_{\lambda_{t-2}}(\epsilon_{(1,t-1)})$.

From these results, the magnitude of $\mathbb{E}_{p(\epsilon)}[\|\mathbf{m}_t - \mathbf{m}_{t-1}\|_2^2]$ can be seen as $\mathcal{O}(\eta_{t-1}^2)$. The same results are obtained for the term $\mathbb{E}_{p(\epsilon)}[\|\mathbf{v}_t - \mathbf{v}_{t-1}\|_2^2]$.

Since $\epsilon \overset{\text{i.i.d.}}{\sim} p(\epsilon) \in \mathbb{R}^d$, the expectation of $\|\epsilon\|_2^2$ is obtained as

$$\mathbb{E}_{p(\epsilon)}\left[\|\epsilon\|_2^2\right] = \mathbb{E}_{p(\epsilon)}[\epsilon_{(1)}^2] + \mathbb{E}_{p(\epsilon)}[\epsilon_{(2)}^2] + \cdots + \mathbb{E}_{p(\epsilon)}[\epsilon_{(d)}^2] = d\mathbb{E}_{p(\epsilon)}[\epsilon_{(1)}^2].$$

If we consider $\mathbb{E}_{p(\epsilon)}[\epsilon_{(1)}^2] \leq \delta (\geq 0)$, $\mathbb{E}_{p(\epsilon)}[\|\mathbf{m}_t - \mathbf{m}_{t-1}\|_2^2] \leq c_1\eta_{t-1}^2$, and $\mathbb{E}_{p(\epsilon)}[\|\mathbf{v}_t - \mathbf{v}_{t-1}\|_2^2] \leq c_2\eta_{t-1}^2$, we can obtain

$$\mathbb{E}_{p(\epsilon)}[\|g_{\lambda_t}(\epsilon) - g_{\lambda_{t-1}}(\epsilon)\|_2^2] \leq K_1(c_1\eta_{t-1}^2 + d\delta c_2\eta_{t-1}^2) = \eta_{t-1}^2 K_1(c_1 + d\delta c_2),$$

where δ, C_1, and C_2 are positive constants.

As $t \to \infty$, we can see that $\mathbb{E}_{p(\epsilon)}[||g_{\lambda_t}(\epsilon) - g_{\lambda_{t-1}}(\epsilon)||_2^2] = \mathcal{O}(\eta_{t-1}^2)$. Furthermore, $\mathbb{V}_t$ is typically similar in magnitude to $\mathbb{E}_{p(\epsilon)}[||g_{\lambda_t}(\epsilon) - g_{\lambda_{t-1}}(\epsilon)||_2^2]$ (Giles, 2015) because

$$\begin{aligned}\mathbb{V}[g_{\lambda_t}(\epsilon) - g_{\lambda_{t-1}}(\epsilon)] &= \mathbb{E}_{p(\epsilon)}\left[\left\|g_{\lambda_t}(\epsilon) - g_{\lambda_{t-1}}(\epsilon)\right\|_2^2\right] - \left\|\mathbb{E}_{p(\epsilon)}\left[g_{\lambda_t}(\epsilon) - g_{\lambda_{t-1}}(\epsilon)\right]\right\|_2^2 \\ &\leq \mathbb{E}_{p(\epsilon)}\left[\left\|g_{\lambda_t}(\epsilon) - g_{\lambda_{t-1}}(\epsilon)\right\|_2^2\right].\end{aligned}$$

Therefore, we find that $\mathbb{V}[g_{\lambda_t}(\epsilon) - g_{\lambda_{t-1}}(\epsilon)] = \mathbb{V}_t = \mathcal{O}(\eta_{t-1}^2)$.

Considering the fact that $\alpha_t = \alpha_0\eta_t \overset{t\to\infty}{\to} 0$, the one-sample variance $\mathbb{V}_t$ becomes 0 asymptotically as iteration proceeds. Thus, the claim is proved. ■

B.4 Proof of Theorem 2

Proof Since the order of variance in gradient estimation via Monte Carlo estimation is $\mathcal{O}(N^{-1})$, we can see the order of one-sample gradient variance at iteration $t = 0$ as $\mathbb{V}_0 = \mathcal{O}(1)$. In addition, from Proposition 1, the order of one-sample gradient variance for $t \geq 1$ is $\mathcal{O}(\eta_{t-1}^2)$. Then, we have the upper bound of the total variance as

$$\sum_{t=0}^{T} N_t^{-1}\mathbb{V}_t \leq N_0^{-1}\kappa + \sum_{t=1}^{T} N_t^{-1}\kappa\eta_{t-1}^2,$$

where κ is a positive constant. As Theorem 1, by considering the constraint optimization problem based on the above upper bound, we have the following optimization problem:

$$\min_{N_t} N_0^{-1}\kappa + \sum_{t=1}^{T} N_t^{-1}\kappa\eta_{t-1}^2 \quad \text{s.t.} \quad \sum_{t=0}^{T} N_t C_t \leq M,$$

where $M = \sum_{t=0}^{T} N_0 C_t$. By solving the above with respect to N_t in the same way as Theorem 1, we obtain $N_t = \frac{M}{\sqrt{\nu d\kappa}+\sqrt{2\nu d\kappa}\sum_{t=1}^{T}\eta_{t-1}}\sqrt{\frac{\kappa\eta_{t-1}^2}{C_t}} = \frac{M}{\sqrt{\nu d}+\sqrt{2\nu d}\sum_{t=1}^{T}\eta_{t-1}}\sqrt{\frac{\eta_{t-1}^2}{C_t}}$ and $N_{t+1} = \frac{\eta_t}{\eta_{t-1}}N_t$ for $t \geq 1$. Since $\eta_0 = 1$, we obtain

$$\begin{aligned} N_{t+1} = \frac{\eta_t}{\eta_{t-1}} \cdot \frac{\eta_{t-1}}{\eta_{t-2}} N_{t-1} &= \frac{\eta_t}{\eta_{t-1}} \cdot \frac{\eta_{t-1}}{\eta_{t-2}} \cdots \frac{\eta_1}{\eta_0} N_1 \\ &= \eta_t N_1. \end{aligned}$$

Furthermore, when $t = 1$, we have

$$\begin{aligned} N_1 = \frac{M}{\sqrt{\nu d}+\sqrt{2\nu d}\sum_{t=1}^{T}\eta_{t-1}}\sqrt{\frac{\eta_0^2}{C_1}} &= \frac{\nu d N_0 + 2\nu d N_0 T}{\sqrt{\nu d}+\sqrt{2\nu d}\sum_{t=1}^{T}\eta_{t-1}}\sqrt{\frac{1}{2\nu d}} \\ &= \frac{2T+1}{\sqrt{2}+2\sum_{t=1}^{T}\eta_{t-1}} N_0. \end{aligned}$$

According to the constraint condition, N_1 should satisfy

$$N_0 C_0 + N_1 C_1 \leq N_0 C_0 + N_0 C_1 \implies N_1 \leq N_0.$$

In addition, since $\eta_{t-1} \leq 1$, we have

$$\frac{2T+1}{\sqrt{2}+2\sum_{t=1}^{T}\eta_{t-1}} N_0 \geq \frac{2T+1}{2T+\sqrt{2}} N_0.$$

From these, we have the following relationship:

$$\frac{2T+1}{2T+\sqrt{2}} N_0 \leq N_1 \leq N_0.$$

Thus, the claim is proved. ■

B.5 Proof of Lemma 2

Proof From Definition 2, since $r_0 = 0$, we can see

$$\sum_{t=0}^{T}(N_t + r_t)C_t = \nu d N_0 + 2\nu d \sum_{t=1}^{T}(N_t + r_t).$$

Then, we obtain

$$\begin{aligned}
\sum_{t=0}^{T} N_t C_t - \sum_{t=0}^{T}(N_t + r_t)C_t &= \nu d N_0 + 2\nu d N_0 T - \nu d N_0 - 2\nu d \sum_{t=1}^{T}(N_t + r_t) \\
&\propto N_0 T - \sum_{t=1}^{T}(N_t + r_t) \\
&= N_0 T - \sum_{t=1}^{T} N_t - \sum_{t=1}^{T} r_t \\
&> N_0 T - \sum_{t=1}^{T} N_t - T \\
&= N_0 T - T - N_1 \sum_{t=1}^{T} \eta_t \\
&= N_0 T - T - N_0 \sum_{t=1}^{T} \eta_t.
\end{aligned}$$

By setting the learning rate scheduler function η_t as $\eta_t = 1$ for $1 \leq t \leq l$, where $l \in \mathbb{Z}$ and $1 \leq l \leq T-1$, we can rearrange the above as

$$N_0 T - T - N_0 \sum_{t=1}^{T} \eta_t = N_0 T - T - N_0 l - N_0 \sum_{t=l+1}^{T} \eta_t.$$

Then, if we set $\eta_t \leq 1 - \frac{2}{N_0}$ for $l+1 \leq t \leq T$, we obtain

$$\begin{aligned}
N_0 T - T - N_0 l - N_0 \sum_{t=l+1}^{T} \eta_t &\geq N_0 T - T - N_0 l - N_0 (T-l)\left(1 - \frac{2}{N_0}\right) \\
&= N_0 T - T - N_0 l - N_0 (T-l) + 2(T-l) \\
&= T - 2l.
\end{aligned}$$

By assuming $T \geq 2$ and taking l as $1 \leq l \leq \lfloor \frac{T}{2} \rfloor$, where $\lfloor x \rfloor = \max\{k \in \mathbb{Z} | k \leq x\}$, we obtain

$$\begin{aligned}
T - 2l &\geq T - 2\left\lfloor \frac{T}{2} \right\rfloor \\
&\geq T - 2 \cdot \frac{T}{2} = 0.
\end{aligned}$$

Thus,

$$\sum_{t=0}^{T} N_t C_t - \sum_{t=0}^{T}(N_t + r_t)C_t > 0,$$

and the claim is proved.

■

B.6 Proof of Lemma 3

Proof According to Proposition 1, the order of $\mathbb{V}_t$ is $\mathcal{O}(\eta_{t-1}^2)$, i.e., $\mathbb{V}_t \leq \kappa_t \eta_{t-1}^2$, where κ_t is a positive constant at the iteration t. Then, we obtain the following inequality:

$$\begin{aligned}
\mathbb{V}[\widehat{\nabla}_{\lambda_t}^{\mathrm{MLRG}} \mathcal{F}(\lambda_t)] &= \alpha_t^{-2} \mathbb{V}[\alpha_t \widehat{\nabla}_{\lambda_t} \mathcal{F}(\lambda_t)] \\
&= \alpha_t^{-2} \mathbb{V}\left[\frac{\eta_t}{\eta_{t-1}} (\lambda_t - \lambda_{t-1}) - \alpha_t N_t^{-1} \sum_{n=1}^{N_t} \left[g_{\lambda_t}(\epsilon_{(n,t)}) - g_{\lambda_{t-1}}(\epsilon_{(n,t)}) \right] \right] \\
&= \mathbb{V}\left[N_t^{-1} \sum_{n=1}^{N_t} \left[g_{\lambda_t}(\epsilon_{(n,t)}) - g_{\lambda_{t-1}}(\epsilon_{(n,t)}) \right] \right] \\
&= N_t^{-1} \mathbb{V}\left[g_{\lambda_t}(\epsilon_{(1,t)}) - g_{\lambda_{t-1}}(\epsilon_{(1,t)}) \right] \leq N_t^{-1} \kappa_t \eta_{t-1}^2.
\end{aligned}$$

Therefore,

$$\mathbb{V}[\widehat{\nabla}_{\lambda_t}^{\mathrm{MLRG}} \mathcal{F}(\lambda_t)] \leq N_t^{-1} \kappa_t \eta_{t-1}^2 = \kappa_t \cdot \eta_{t-1}^2 N_t^{-1} = \mathcal{O}(\eta_{t-1}^2 N_t^{-1}),$$

and the claim is proved. ■

B.7 Proof of Theorem 3

Proof Accorging to Assumption 5, we have $\mathcal{F}(\lambda) \leq \mathcal{F}(\bar{\lambda}) + \nabla_{\bar{\lambda}} \mathcal{F}(\bar{\lambda})^\top (\lambda - \bar{\lambda}) + \frac{1}{2} K_2 ||\lambda - \bar{\lambda}||_2^2, \forall \lambda, \bar{\lambda}$. By using the SGD update rule, we obtain $\lambda_{t+1} - \lambda_t = -\alpha_t \hat{g}_{\lambda_t}(\epsilon_{1:N})$. Thus, when we set $\lambda = \lambda_{t+1}$ and $\bar{\lambda} = \lambda_t$, this assumption can be expressed as

$$\begin{aligned}
\mathcal{F}(\lambda_{t+1}) - \mathcal{F}(\lambda_t) &\leq \nabla_{\lambda_t} \mathcal{F}(\lambda_t)^\top (\lambda_{t+1} - \lambda_t) + \frac{1}{2} K_2 ||\lambda_{t+1} - \lambda_t||_2^2 \\
&= -\alpha_t \nabla_{\lambda_t} \mathcal{F}(\lambda_t)^\top \hat{g}_{\lambda_t}(\epsilon_{1:N_t}) + \frac{\alpha_t^2 K_2}{2} ||\hat{g}_{\lambda_t}(\epsilon_{1:N})||_2^2.
\end{aligned}$$

By taking expectation in the above with respect to $\epsilon_{1:N} \sim p(\epsilon)$, we obtain

$$\mathbb{E}_{p(\epsilon_{1:N})}[\mathcal{F}(\lambda_{t+1}) - \mathcal{F}(\lambda_t)] \leq -\alpha_t \nabla_{\lambda_t} \mathcal{F}(\lambda_t)^\top \mathbb{E}_{p(\epsilon_{1:N})}[\hat{g}_{\lambda_t}(\epsilon_{1:N})] + \frac{\alpha_t^2 K_2}{2} \mathbb{E}_{p(\epsilon_{1:N})}\left[\|\hat{g}_{\lambda_t}(\epsilon_{1:N})\|_2^2 \right].$$

Since $\mathbb{E}_{p(\epsilon_{1:N})}[\hat{g}_{\lambda_t}(\epsilon_{1:N})] = \nabla_{\lambda_t} \mathcal{F}(\lambda_t)$ and $\mathbb{E}_{p(\epsilon_{1:N})}[||\hat{g}_{\lambda_t}(\epsilon_{1:N})||_2^2] = \mathbb{V}[\hat{g}_{\lambda_t}(\epsilon_{1:N})] + ||\mathbb{E}_{p(\epsilon_{1:N})}[\hat{g}_{\lambda_t}(\epsilon_{1:N})]||_2^2$, we obtain

$$\begin{aligned}
&\mathbb{E}_{p(\epsilon_{1:N})}[\mathcal{F}(\lambda_{t+1}) - \mathcal{F}(\lambda_t)] \\
&\quad \leq -\alpha_t ||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2 + \frac{\alpha_t^2 K_2}{2} \left(\mathbb{V}[\hat{g}_{\lambda_t}(\epsilon_{1:N})] + \left\| \mathbb{E}_{p(\epsilon_{1:N})}[\hat{g}_{\lambda_t}(\epsilon_{1:N})] \right\|_2^2 \right).
\end{aligned}$$

Again, since $\mathbb{E}_{p(\epsilon_{1:N})}[\hat{g}_{\lambda_t}(\epsilon_{1:N})] = \nabla_{\lambda_t} \mathcal{F}(\lambda_t)$, we can rewrite the above equation as

$$\mathbb{E}_{p(\epsilon_{1:N})}[\mathcal{F}(\lambda_{t+1}) - \mathcal{F}(\lambda_t)] \leq \frac{\alpha_t^2 K_2}{2} \mathbb{V}[\hat{g}_{\lambda_t}(\epsilon_{1:N})] + \left(\frac{\alpha_t^2 K_2}{2} - \alpha_t \right) ||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2.$$

By summing $t = 1, 2, \ldots, T$ and taking the total expectation, we obtain

$$\mathbb{E}[\mathcal{F}(\lambda_T) - \mathcal{F}(\lambda_1)] \leq \frac{K_2}{2} \sum_{t=1}^{T} \alpha_t^2 \mathbb{E}\Big[\mathbb{V}[\hat{g}_{\lambda_t}(\epsilon_{1:N})]\Big] + \sum_{t=1}^{T} \left(\frac{\alpha_t^2 K_2}{2} - \alpha_t\right) \mathbb{E}\Big[||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2\Big].$$

Since $\mathcal{F}(\lambda^*) - \mathcal{F}(\lambda_1) \leq \mathbb{E}[\mathcal{F}(\lambda_T) - \mathcal{F}(\lambda_1)]$, where λ_1 is deterministic and λ^* is the optimal parameter, we obtain the following inequality by dividing the inequality by $A_T = \sum_{t=1}^{T} \alpha_t$:

$$\frac{1}{A_T}[\mathcal{F}(\lambda^*) - \mathcal{F}(\lambda_1)] \leq \frac{K_2}{2A_T} \sum_{t=1}^{T} \alpha_t^2 \mathbb{E}\Big[\mathbb{V}[\hat{g}_{\lambda_t}(\epsilon_{1:N})]\Big] + \frac{1}{A_T} \sum_{t=1}^{T} \left(\frac{\alpha_t^2 K_2}{2} - \alpha_t\right) \mathbb{E}\Big[||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2\Big].$$

Therefore, we can obtain

$$\begin{aligned}
&\frac{1}{A_T} \sum_{t=1}^{T} \alpha_t \mathbb{E}\Big[||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2\Big] \\
&\qquad \leq \frac{1}{A_T}[\mathcal{F}(\lambda_1) - \mathcal{F}(\lambda^*)] + \frac{K_2}{2A_T} \sum_{t=1}^{T} \alpha_t^2 \mathbb{E}\Big[||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2\Big] + \frac{K_2}{2A_T} \sum_{t=1}^{T} \alpha_t^2 \mathbb{E}\Big[\mathbb{V}[\hat{g}_{\lambda_t}(\epsilon_{1:N})]\Big].
\end{aligned}$$

If we estimate the gradient values using MC samples, we can see that $\mathbb{V}[\hat{g}_{\lambda_t}(\epsilon_{1:N})] \leq \kappa N^{-1}$ for some positive constant κ; therefore, we obtain

$$\begin{aligned}
&\frac{1}{A_T} \sum_{t=1}^{T} \alpha_t \mathbb{E}\Big[||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2\Big] \\
&\qquad \leq \frac{1}{A_T}[\mathcal{F}(\lambda_1) - \mathcal{F}(\lambda^*)] + \frac{K_2}{2A_T} \sum_{t=1}^{T} \alpha_t^2 \mathbb{E}\Big[||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2\Big] + \frac{K_2}{2A_T} \sum_{t=1}^{T} \alpha_t^2 \mathbb{E}\Big[\mathbb{V}[\hat{g}_{\lambda_t}(\epsilon_{1:N})]\Big] \\
&\qquad \leq \frac{1}{A_T}[\mathcal{F}(\lambda_1) - \mathcal{F}(\lambda^*)] + \frac{K_2}{2A_T} \sum_{t=1}^{T} \alpha_t^2 \mathbb{E}\Big[||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2\Big] + \frac{\kappa \cdot K_2}{2A_T N} \sum_{t=1}^{T} \alpha_t^2 \\
&\qquad = \frac{1}{A_T}[\mathcal{F}(\lambda_1) - \mathcal{F}(\lambda^*)] + \frac{\alpha_0^2 K_2}{2A_T} \sum_{t=1}^{T} \eta_t^2 \left(\mathbb{E}\Big[||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2\Big] + \frac{\kappa}{N}\right).
\end{aligned}$$

The last term is derived from $\alpha_t = \alpha_0 \cdot \eta_t$. When estimating the gradient value using RQMC samples, since the order of variance is $\mathcal{O}(N^{-2})$, we can obtain the same as above as follows:

$$\frac{1}{A_T} \sum_{t=1}^{T} \alpha_t \mathbb{E}\Big[||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2\Big] \leq \frac{1}{A_T}[\mathcal{F}(\lambda_1) - \mathcal{F}(\lambda^*)] + \frac{\alpha_0^2 K_2}{2A_T} \sum_{t=1}^{T} \eta_t^2 \left(\mathbb{E}\Big[||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2\Big] + \frac{\kappa}{N^2}\right).$$

Thus, the claim is proved. ■

B.8 Proof of Theorem 3 for the MLMC-based Method

Proof By taking the same result from the above proof of Theorem 3, we obtain

$$\mathcal{F}(\lambda_{t+1}) - \mathcal{F}(\lambda_t) \leq -\alpha_t \nabla_{\lambda_t} \mathcal{F}(\lambda_t)^\top \hat{g}_{\lambda_t}(\epsilon_{1:N_t}) + \frac{K_2}{2} || - \alpha_t \hat{g}_{\lambda_t}(\epsilon_{1:N_t})||_2^2.$$

By taking the expectation with respect to $\epsilon_{1:N} \sim p(\epsilon)$, we obtain

$$\begin{aligned}
&\mathbb{E}_{p(\epsilon_{1:N_t})}[\mathcal{F}(\lambda_{t+1}) - \mathcal{F}(\lambda_t)] \\
&\quad \leq -\alpha_t \nabla_{\lambda_t} \mathcal{F}(\lambda_t)^\top \mathbb{E}_{p(\epsilon_{1:N_t})}[\hat{g}_{\lambda_t}(\epsilon_{1:N_t})] + \frac{K_2}{2} \mathbb{E}_{p(\epsilon_{1:N})} \left[|| - \alpha_t \hat{g}_{\lambda_t}(\epsilon_{1:N_t})||_2^2 \right].
\end{aligned}$$

Using the fact that $\mathbb{E}_{p(\epsilon_{1:N})}[|| - \alpha_t \hat{g}_{\lambda_t}(\epsilon_{1:N})||_2^2] = \mathbb{V}[-\alpha_t \hat{g}_{\lambda_t}(\epsilon_{1:N})] + ||\mathbb{E}_{p(\epsilon_{1:N})}[-\alpha_t \hat{g}_{\lambda_t}(\epsilon_{1:N})]||_2^2$ and $\mathbb{E}_{p(\epsilon_{1:N_t})}[\hat{g}_{\lambda_t}(\epsilon_{1:N_t})] = \nabla_{\lambda_t} \mathcal{F}(\lambda_t)$, we obtain

$$\begin{aligned}
&\mathbb{E}_{p(\epsilon_{1:N_t})}[\mathcal{F}(\lambda_{t+1}) - \mathcal{F}(\lambda_t)] \\
&\quad \leq -\alpha_t ||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2 + \frac{K_2}{2} \left(\mathbb{V}[-\alpha_t \hat{g}_{\lambda_t}(\epsilon_{1:N_t})] + \alpha_t^2 ||\mathbb{E}_{p(\epsilon_{1:N_t})}[\hat{g}_{\lambda_t}(\epsilon_{1:N_t})]||_2^2 \right). \quad (22)
\end{aligned}$$

According to Lemma 1, we obtain $-\alpha_t \hat{g}_{\lambda_t}(\epsilon_{1:N_t}) = \frac{\eta_t}{\eta_{t-1}}(\lambda_t - \lambda_{t-1}) - \alpha_t \hat{g}'_{\lambda_t}(\epsilon_{1:N_t})$, where $\hat{g}'_{\lambda_t}(\epsilon_{1:N_t}) = N_t^{-1} \sum_{n=1}^{N_t} [g_{\lambda_t}(\epsilon_{(n,t)}) - g_{\lambda_{t-1}}(\epsilon_{(n,t)})]$. By using this equation and the result of Lemma 3, we obtain

$$\mathbb{V}[-\alpha_t \hat{g}_{\lambda_t}(\epsilon_{1:N_t})] = \mathbb{V}\left[\frac{\eta_t}{\eta_{t-1}}(\lambda_t - \lambda_{t-1}) - \alpha_t \hat{g}'_{\lambda_t}(\epsilon_{1:N_t}) \right] = \alpha_t^2 \mathbb{V}[\hat{g}'_{\lambda_t}(\epsilon_{1:N_t})] \leq \alpha_t^2 \cdot \kappa \eta_{t-1}^2 N_t^{-1}.$$

Therefore, Eq. 22 can be rewritten as

$$\begin{aligned}
&\mathbb{E}_{p(\epsilon_{1:N_t})}[\mathcal{F}(\lambda_{t+1}) - \mathcal{F}(\lambda_t)] \\
&\quad \leq -\alpha_t ||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2 + \frac{K_2}{2} \left(\mathbb{V}[-\alpha_t \hat{g}_{\lambda_t}(\epsilon_{1:N_t})] + \alpha_t^2 ||\mathbb{E}_{p(\epsilon_{1:N_t})}[\hat{g}_{\lambda_t}(\epsilon_{1:N_t})]||_2^2 \right) \\
&\quad \leq -\alpha_t ||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2 + \frac{K_2}{2} \left(\alpha_t^2 \cdot \kappa \eta_{t-1}^2 N_t^{-1} + \alpha_t^2 ||\mathbb{E}_{p(\epsilon_{1:N_t})}[\hat{g}_{\lambda_t}(\epsilon_{1:N_t})]||_2^2 \right) \\
&\quad = -\alpha_t ||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2 + \frac{\alpha_t^2 K_2}{2} \left(\kappa \eta_{t-1}^2 N_t^{-1} + ||\mathbb{E}_{p(\epsilon_{1:N_t})}[\hat{g}_{\lambda_t}(\epsilon_{1:N_t})]||_2^2 \right).
\end{aligned}$$

Using the fact that $\mathbb{E}_{p(\epsilon_{1:N})}[\hat{g}_{\lambda_t}(\epsilon_{1:N})] = \nabla_{\lambda_t} \mathcal{F}(\lambda_t)$, we can rewrite the above equation as

$$\mathbb{E}_{p(\epsilon_{1:N_t})}[\mathcal{F}(\lambda_{t+1}) - \mathcal{F}(\lambda_t)] \leq \frac{\kappa K_2 \alpha_t^2}{2N_t} \cdot \eta_{t-1}^2 + \left(\frac{\alpha_t^2 K_2}{2} - \alpha_t \right) ||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2.$$

By summing for $t = 1, 2, \ldots, T$ and taking the total expectation, we obtain

$$\mathbb{E}[\mathcal{F}(\lambda_T) - \mathcal{F}(\lambda_1)] \leq \frac{\kappa K_2}{2} \sum_{t=1}^{T} \frac{\alpha_t^2}{N_t} \eta_{t-1}^2 + \sum_{t=1}^{T} \left(\frac{\alpha_t^2 K_2}{2} - \alpha_t \right) \mathbb{E}\left[||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2 \right].$$

From the fact that $\mathcal{F}(\lambda^*) - \mathcal{F}(\lambda_1) \leq \mathbb{E}[\mathcal{F}(\lambda_T) - \mathcal{F}(\lambda_1)]$, we obtain the following inequality by dividing the inequality by $A_T = \sum_{t=1}^T \alpha_t$:

$$\frac{1}{A_T}[\mathcal{F}(\lambda^*) - \mathcal{F}(\lambda_1)] \leq \frac{\kappa K_2}{2A_T} \sum_{t=1}^T \frac{\alpha_t^2}{N_t} \eta_{t-1}^2 + \frac{1}{A_T} \sum_{t=1}^T \left(\frac{\alpha_t^2 K_2}{2} - \alpha_t \right) \mathbb{E}\left[||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2 \right].$$

Therefore, we can obtain

$$\begin{aligned}
&\frac{1}{A_T} \sum_{t=1}^T \alpha_t ||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2 \\
&\quad \leq \frac{1}{A_T}[\mathcal{F}(\lambda_1) - \mathcal{F}(\lambda^*)] + \frac{K_2}{2A_T} \sum_{t=1}^T \alpha_t^2 \mathbb{E}\left[||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2 \right] + \frac{\kappa K_2}{2A_T} \sum_{t=1}^T \frac{\alpha_t^2}{N_t} \eta_{t-1}^2 \\
&\quad = \frac{1}{A_T}[\mathcal{F}(\lambda_1) - \mathcal{F}(\lambda^*)] + \frac{\alpha_0^2 K_2}{2A_T} \sum_{t=1}^T \eta_t^2 \left(\mathbb{E}\left[||\nabla_{\lambda_t} \mathcal{F}(\lambda_t)||_2^2 \right] + \frac{\kappa}{N_t} \eta_{t-1}^2 \right),
\end{aligned}$$

and the claim is proved. ■

B.9 Proof of Theorem 4

Proof Firstly, we focus on the MC- and RQMC-based methods. Because of the definition of SNR and $\mathbb{E}_{p(\epsilon)}[\hat{g}_\lambda(\epsilon_{1:N})] = \nabla_{\lambda_t} \mathcal{F}(\lambda_t)$, we obtain

$$\mathrm{SNR}(\lambda) = \frac{\|\mathbb{E}_{p(\epsilon_{1:N})}[\hat{g}_\lambda(\epsilon_{1:N})]\|_2^2}{\sqrt{\mathbb{V}[\hat{g}_\lambda(\epsilon_{1:N})]}} = \frac{\|\nabla_\lambda \mathcal{F}(\lambda)\|_2^2}{\sqrt{\mathbb{V}[\hat{g}_\lambda(\epsilon_{1:N})]}}.$$

Suppose that Assumptions 2–4 hold, the expectation and variance of the gradient estimator are nonzero, and the variances of $\hat{g}_\lambda(\epsilon_{1:N})$ are also nonzero. Then, we have the upper bounds of gradient variance for the MC- and RQMC-based methods as, for all λ_t, $\mathbb{V}[\hat{g}_{\lambda_t}] \leq \kappa N^{-1}$ and $\mathbb{V}[\hat{g}_{\lambda_t}] \leq \kappa N^{-2}$, respectively, where κ is a positive constant. By using the order of gradient variance in the above, we obtain the following lower bounds:

$$\mathrm{SNR}(\lambda_t) = \frac{\|\nabla_{\lambda_t} \mathcal{F}(\lambda_t)\|_2^2}{\sqrt{\mathbb{V}[\hat{g}_{\lambda_t}(\epsilon_{1:N})]}} \geq \begin{cases} \frac{\|\nabla_{\lambda_t} \mathcal{F}(\lambda_t)\|_2^2}{\sqrt{\kappa N^{-1}}}, \\ \frac{\|\nabla_{\lambda_t} \mathcal{F}(\lambda_t)\|_2^2}{\sqrt{\kappa N^{-2}}}, \end{cases} = \begin{cases} \frac{\|\nabla_{\lambda_t} \mathcal{F}(\lambda_t)\|_2^2}{\sqrt{\kappa}} \cdot \sqrt{N} & \text{(MC)}, \\ \frac{\|\nabla_{\lambda_t} \mathcal{F}(\lambda_t)\|_2^2}{\sqrt{\kappa}} \cdot N & \text{(RQMC)}. \end{cases}$$

Secondly, we show the SNR bound of our method. For our method, the SNR can be expressed as

$$\mathrm{SNR}(\lambda_t) = \frac{\|\mathbb{E}_{p(\epsilon_{1:N})}[\hat{g}_\lambda(\epsilon_{1:N})]\|_2^2}{\sqrt{\mathbb{V}[\hat{g}_\lambda(\epsilon_{1:N})]}} = \frac{\|\nabla_{\lambda_t} \mathcal{F}(\lambda_t)\|_2^2}{\sqrt{\alpha_t^{-2} \mathbb{V}[\alpha_t \hat{g}_\lambda(\epsilon_{1:N})]}}.$$

According to the proof of Theorem 3 for the MLMC-based method,

$$\mathbb{V}[\alpha_t \hat{g}_\lambda(\epsilon_{1:N})] = \mathbb{V}[-\alpha_t \hat{g}_\lambda(\epsilon_{1:N})] \leq \alpha_t^2 \cdot \kappa \eta_{t-1}^2 N_t^{-1}.$$

Therefore, we obtain

$$\mathrm{SNR}(\lambda_t) \geq \frac{\|\nabla_{\lambda_t}\mathcal{F}(\lambda_t)\|_2^2}{\sqrt{\alpha_t^{-2}\alpha_t^2 \cdot \kappa\eta_{t-1}^2 N_t^{-1}}} = \frac{\|\nabla_{\lambda_t}\mathcal{F}(\lambda_t)\|_2^2}{\sqrt{\kappa\eta_{t-1}^2 N_t^{-1}}} = \frac{\|\nabla_{\lambda_t}\mathcal{F}(\lambda_t)\|_2^2}{\sqrt{\kappa}} \cdot \frac{\sqrt{N_t}}{\eta_{t-1}}.$$

Thus, the claim is proved. ■

Appendix C. Unbounded Case in Assumption 3

Assumption 3 does not always hold because $\mathcal{T}(\epsilon;\lambda)$ itself is not bounded. To overcome this problem, the proximal operator is useful. The proximal operator is an operator associated with a closed convex function f from a Hilbert space $\mathcal{X}$ to $[-\infty,\infty]$, and it is defined as

$$\mathrm{prox}_f(x) = \underset{u\in\mathcal{X}}{\mathrm{argmin}} \left(f(u) + \frac{1}{2}\|u-x\|_2^2 \right).$$

In optimization, the proximal operator has several useful properties such as its firm nonexpansiveness as follow:

$$\|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|_2^2 \leq (\mathrm{prox}_f(x) - \mathrm{prox}_f(y))^\top (x-y),$$

where $\forall x, y \in \mathcal{X}$.

We will apply this to the case where $\mathcal{T}(\epsilon;\lambda)$ becomes too large to satisfy Assumption 3 in our method. We set f as the following indicator function for some set S:

$$\mathrm{I}_S(\mathcal{T}(\epsilon;\lambda)) = \begin{cases} 0 & (\mathcal{T}(\epsilon;\lambda) \in S), \\ +\infty & (\text{otherwise}), \end{cases}$$

where $\mathrm{I}_S(\mathcal{T}(\epsilon;\lambda))$ is closed and convex if S is a closed convex set. From this setting, the proximal operator $f = \mathrm{I}_S$ is the Euclidian projection $P(\cdot)$ on S:

$$\mathrm{prox}_{\mathrm{I}_S}(\mathcal{T}(\epsilon;\lambda)) = \underset{u\in S}{\mathrm{argmin}} \|u - \mathcal{T}(\epsilon;\lambda)\|_2^2 = P_S(\mathcal{T}(\epsilon;\lambda)).$$

From the above and firm nonexpansiveness of the proximal operator, if $\mathcal{T}^+(\epsilon;\lambda) = \mathrm{prox}_{\mathrm{I}_S}(\mathcal{T}(\epsilon;\lambda))$ and $\mathcal{T}^+(\epsilon;\bar{\lambda}) = \mathrm{prox}_{\mathrm{I}_S}(\mathcal{T}(\epsilon;\bar{\lambda}))$, then

$$\|\mathcal{T}^+(\epsilon;\lambda) - \mathcal{T}^+(\epsilon;\bar{\lambda})\|_2^2 \leq (\mathcal{T}^+(\epsilon;\lambda) - \mathcal{T}^+(\epsilon;\bar{\lambda}))^\top (\mathcal{T}(\epsilon;\lambda) - \mathcal{T}(\epsilon;\bar{\lambda}))$$

is fulfilled. This implies that

$$\|\mathcal{T}^+(\epsilon;\lambda) - \mathcal{T}^+(\epsilon;\bar{\lambda})\|_2^2 \leq \|\mathcal{T}(\epsilon;\lambda) - \mathcal{T}^+(\epsilon;\bar{\lambda})\|_2^2$$

from the Cauchy-–Schwarz inequality. It means that $\mathcal{T}^+(\epsilon;\lambda)$ is 1-Lipschitz continuous and therefore bounded.

By using the proximal operator, we can obtain the samples that do not violate Assumption 3. Therefore, all the theorems and lemmas in this paper hold in a unbounded case of $\mathcal{T}(\epsilon;\lambda)$. The MLMCVI algorithm in a unbounded case is shown in Algorithm 2.

Algorithm 2 Multilevel Monte Carlo Variational Inference in Unbounded Case of $\mathcal{T}(\epsilon;\lambda)$

Require: Data $\mathbf{x}$, random variable $\epsilon \sim p(\epsilon)$, transform $\mathbf{z} = \mathcal{T}(\epsilon;\lambda)$, model $p(\mathbf{x},\mathbf{z})$, variational family $q(\mathbf{z}|\lambda)$
Ensure: Variational parameter λ^*
Initialize: N_0, λ_0, α_0, the hyperparameter of η, and a convex set S
for $t = 0$ to T **do**
  if $t = 0$ **then**
    $\epsilon_n \sim p(\epsilon)$ $(n = 1, 2, \ldots, N_0)$ ◁ sampling ϵ
    $\mathcal{T}^+(\epsilon_n;\lambda_0) = \mathrm{prox}_{\mathrm{I}_S}(\mathcal{T}(\epsilon_n;\lambda_0))$ ◁ check $\mathcal{T}(\epsilon;\lambda_0)$ value
    $\hat{g}_{\lambda_0}(\epsilon_{1:N_0}) = N_0^{-1}\sum_{n=1}^{N_0} g_{\lambda_0}(\epsilon_n)$ ◁ calc. RG estimator
    $\lambda_1 = \lambda_0 - \alpha_0 \hat{g}_{\lambda_0}(\epsilon_{1:N_0})$ ◁ grad-update
  else
    estimate N_t using $N_t = \lceil \eta_{t-1} N_0 \rceil$
    sampling one ϵ for sample size estimation
    $\epsilon_n \sim p(\epsilon)$ $(n = 1, 2, \ldots, N_t)$ ◁ sampling ϵ
    $\mathcal{T}^+(\epsilon_n;\lambda_t) = \mathrm{prox}_{\mathrm{I}_S}(\mathcal{T}(\epsilon_n;\lambda_t))$ ◁ check $\mathcal{T}(\epsilon;\lambda_t)$ value
    $\hat{g}'_{\lambda_t}(\epsilon_{1:N_t}) = N_t^{-1}\sum_{n=1}^{N_t}[g_{\lambda_t}(\epsilon_{(n,t)}) - g_{\lambda_{t-1}}(\epsilon_{(n,t)})]$ ◁ calc. Multilevel term
    $\lambda_{t+1} = \lambda_t + \frac{\eta_t}{\eta_{t-1}}(\lambda_t - \lambda_{t-1}) - \alpha_t \hat{g}'_{\lambda_t}(\epsilon_{1:N_t})$ ◁ grad-update
    if λ_{t+1} has converged to λ^* **then**
      break
    end if
  end if
end for
return λ^*

One possible problem with artificially truncating the value of $\mathcal{T}(\epsilon;\lambda)$ is that the gradient estimator can be biased because multiple values of the samples ϵ may be mapped to the same bounded value of $\mathcal{T}(\epsilon;\lambda)$. However, in practice, we can reduce this bias by taking a sufficiently large closed convex set S so that we can take $u \in S$ more flexible. Furthermore, the truncation through the proximal operator can be seen as the "gradient clipping" technique that has been empirically successful, especially in deep learning context (Mikolov, 2012; Pascanu et al., 2013; Kanai et al., 2017; Belghazi et al., 2018). From these empirical successes, the gradient truncation through a proximal operator is one realistic way to ensure that Assumption 3 is satisfied. A summary of the gradient clipping can be seen in Section 10.11.1 of the paper by Goodfellow et al. (2016).

Theoretical analyses of the properties of gradient clipping, such as analysis of bias due to truncation by the proximal operator, is beyond the scope of our paper. However, such theoretical analyses are important for improving the performance of optimization through gradient clipping. Research on why gradient clipping works well has recently been conducted, e.g., Zhang et al. (2020), and such research on the theoretical exploration of gradient clipping is beginning to attract attention. Thus, it may be very interesting for future research to discuss how the bias due to gradient clipping, such as truncation, affects stochastic optimization in the MCVI context.

Appendix D. Learning Rate Scheduler and Sample Size Estimation

From Algorithm 1, we can estimate the number of samples N_t as $N_{t+1} = \lceil \eta_t N_1 \rceil$ and $N_0 = N_1$ from Theorem 2 and Lemma 2. Therefore, its estimation scheme varies with the learning rate scheduler function η_{t-1}. There are three major learning rate schedulers: time-based decay, step-based decay, and exponential decay defined as follows.

Definition 3 (Time-based Decay Function) *Time-based decay function η_t is defined as $\eta_t = \frac{1}{1+\beta t}$, where β is the parameter of the degree of decay.*

Definition 4 (Step-based Decay Function) *Step-based decay function η_t is defined as $\eta_t = \beta^{\lfloor \frac{t}{r} \rfloor}$, where β is the parameter of the degree of decay, and r is the drop-rate parameter. Here, we denote the function $\lfloor x \rfloor$ as $\lfloor x \rfloor = \max\{k \in \mathbb{Z} | k \leq x\}$.*

Definition 5 (Exponential Decay Function) *Exponential decay function η_t is defined as $\eta_t = \exp(-\beta t)$, where β is the parameter of the degree of decay.*

In these functions, N_t for $t \geq 1$ is estimated as follows:

- Time-based decay: $N_t = \lceil \frac{1}{1+\beta(t-1)} N_0 \rceil$,
- Step-based decay: $N_t = \lceil \beta^{\lfloor \frac{t-1}{r} \rfloor} N_0 \rceil$,
- Exponential decay: $N_t = \lceil \exp(-\beta(t-1)) N_0 \rceil$,

where β and r are the decay and drop-rate parameters, respectively.

Appendix E. Memory Space Cost and Time Cost for the MLRG Estimator

In this section, we discuss how our method affects the memory cost of gradient estimation. Since the MC- and RQMC-based methods estimate the ELBO or the variational free energy using the current parameters and evaluate its stochastic gradient, the required cost of memory space is $\mathcal{O}(d)$ per iteration, where d is the dimension of the parameter. Furthermore, the time cost can be seen as $\mathcal{O}(N \times d)$, where N is the sample size for gradient estimation.

In contrast, in our method, we should keep the old parameter and estimate the old gradient per iteration; hence, we can see that the required cost of memory space is $\mathcal{O}(2d)$. In addition, the time cost of our method can be seen as $\mathcal{O}(2 \times N_t \times d)$. We summarize the memory space cost and the time cost of each method in Table 2.

It may seem that the time cost of our method is too large compared with that of the baseline methods. However, as shown in Figure 7, the time cost of our method becomes smaller than that of the baseline methods as the optimization proceeds because our method decreases the sample size for gradient estimation using a learning rate scheduler η.

Therefore, the hyperparameter optimization of the learning rate scheduler is important in our method. If the sample size is reduced too fast, the time cost will be decreased quickly; however, the inference may be affected. On the other hand, if the sample size is reduced too slowly, the estimation can be stable, but the time cost will be large through optimization. Unfortunately, theoretically deriving the optimal hyperparameters is an open problem in stochastic optimization. One practical way to optimize a hyperparameter of the learning rate scheduler is using a hyperparameter optimization tool such as optuna (Akiba et al., 2019).

	MC or RQMC	MLMC
Time cost	$\mathcal{O}(N \times d)$	$\mathcal{O}(2 \times N_t \times d)$
Memory space	$\mathcal{O}(d)$	$\mathcal{O}(2 \times d)$

Table 2: Summary of time cost and cost of memory space.

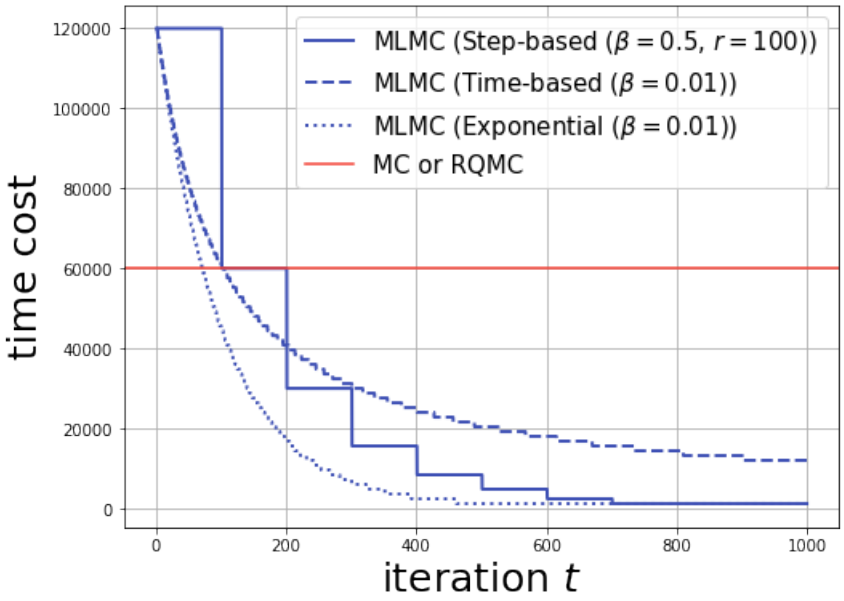

Figure 7: Time cost vs. iteration plot.

Appendix F. Details of Models in Experiments

In this section, we show the details of the model generative process and hyperparameter settings used in our experiments.

F.1 Hierarchical Linear Regression

We applied hierarchical linear regression to toy data generated from the same generating process of this model. Here, we set a Gaussian hyperprior on μ' and lognormal hyperpriors on the variance of intercepts σ' and the noise ϵ.

The generative process of this model is as follows.

$$
\begin{aligned}
\mu' &\sim \mathcal{N}(0, 10^2), && \text{weight hyperprior} \\
\sigma' &\sim \text{LogNormal}(0.5), && \text{weight hyperprior} \\
\epsilon &\sim \text{LogNormal}(0.5), && \text{noise} \\
\mathbf{b}_i &\sim \mathcal{N}(\mu', \sigma'), && \text{weights} \\
y_i &\sim \mathcal{N}(\mathbf{x}_i^\top \mathbf{b}_i, \epsilon). && \text{output distribution}
\end{aligned}
$$

We set $I = 100$ and $k = 10$, where k denotes the dimension of the data x_i and I is the number of observations. In this setting, the dimension of the entire parameter space is

$d = I \times k + k + 2 = 1012$, and this model is approximated by a variational diagonal Gaussian distribution.

We optimized the variational free energy of the MC- and RQMC-based methods by using the Adam optimizer (Kingma and Ba, 2015) and that of the MLMC-based method by using the SGD optimizer with a learning rate scheduler η with 100 initial MC or RQMC samples. We compared the empirical variance and SNR of these methods by using 100 or 10 initial MC or RQMC samples for inference. In the optimization step, we used η as the step-decay function. The initial learning rate ($\in [10^{-3}, 0.5]$) and the hyperparameters of the learning rate scheduler, β ($\in [0.1, 1]$) and r ($\in [10, 500]$), were optimized through the Tree-structured Parzen Estimator (TPE) sampler in `optuna` (Akiba et al., 2019) with 50 trials, where β and r are the decay and drop-rate parameter, respectively.

F.2 Bayesian Logistic Regression

Binary Classification: We applied Bayesian Logistic Regression to the breast cancer dataset in the UCI Machine Learning Repository for a binary classification task. Here, we set a standard Gaussian hyperprior on μ' and an inverse gamma hyperprior (weak information prior) on the variance of weights σ'.

The generative process of this model is as follows.

$$
\begin{aligned}
&\sigma' \sim \mathrm{Gamma}(0.5, 0.5), && \text{weight hyperprior}\\
&\mu' \sim \mathcal{N}(0, 1), && \text{weight hyperprior}\\
&\mathbf{z}_i \sim \mathcal{N}(\mu', 1/\sigma'), && \text{weights}\\
&\sigma(x) = \frac{1}{1 + \exp(-x)}, && \text{Sigmoid function}\\
&y_i \sim \mathrm{Bernoulli}(\sigma(\mathbf{x}_i^\top \mathbf{z}_i)). && \text{output distributions}
\end{aligned}
$$

In these settings, the dimension of the entire parameter space is $d = 11$, and this model is approximated by a variational diagonal Gaussian distribution.

To optimize the variational free energy, we used the Adam optimizer for the MC- and RQMC-based methods and the SGD optimizer with a learning rate scheduler η for the MLMC-based method. We used 100 initial MC or RQMC samples for gradient estimation.

We compared the empirical variance and SNR of these methods by using 100 or 10 initial MC or RQMC samples for inference. In the optimization step, we used η as the step-decay function. The initial learning rate ($\in [10^{-5}, 10^{-2}]$) and the hyperparameters of the learning rate scheduler, β ($\in [0.1, 1]$) and r ($\in [10, 500]$), were optimized through the Tree-structured Parzen Estimator (TPE) sampler in `optuna` (Akiba et al., 2019) with 50 trials, where β and r are the decay and drop-rate parameter, respectively.

Multilabel Classification: We also applied this model to the Fashion-MNIST dataset for a multilabel classification task. We set a standard Gaussian hyperprior on μ' and an inverse gamma hyperprior (weak information prior) on the variance of weights σ'. Thus, the

generative process of this model is as follows.

$$\begin{aligned}
&\sigma' \sim \text{Gamma}(0.5, 0.5), && \text{weight hyperprior}\\
&\mu' \sim \mathcal{N}(0, 1), && \text{weight hyperprior}\\
&\mathbf{z}_i \sim \mathcal{N}(\mu', 1/\sigma'), && \text{weights}\\
&\sigma(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}, && \text{Softmax function}\\
&y \sim \text{Categorical}(\sigma(\phi(\mathbf{x}_i^\top \mathbf{z}_i))). && \text{output distributions}
\end{aligned}$$

In these settings, the dimension of the entire parameter space is $d = 7852$, and this model is also approximated by a variational diagonal Gaussian distribution.

We optimized the variational free energy of the MC-, RQMC-, and MLMC-based methods by using the SGD optimizer with a learning rate scheduler η. We used 100 initial MC or RQMC samples for gradient estimation. In the optimization step, we used η as the step-decay function and set the hyperparameter $\{\beta, r\}$ for sample size estimation to $\{0.5, 100\}$. Finally, we set the initial learning rate as 0.01, 0.005, or 0.001.

F.3 Bayesian Neural Network Regression

We applied a BNN regression model to the wine-quality-red dataset, which is included the wine-quality dataset in the UCI Machine Learning Repository.

The network consists of a 50-unit hidden layer with ReLU activations. In addition, we set a normal prior over each weight and placed an inverse Gamma hyperprior over each weight prior, and we also set an inverse Gamma hyperprior to the observed variance.

The generative process of this model is as follows.

$$\begin{aligned}
&\alpha \sim \text{Gamma}(1., 0.1), && \text{weight hyperprior}\\
&\tau \sim \text{Gamma}(1., 0.1), && \text{noise hyperprior}\\
&w_i \sim \mathcal{N}(0, 1/\alpha), && \text{weights}\\
&y \sim \mathcal{N}(\phi(\mathbf{x}, \mathbf{w}), 1/\tau). && \text{output distributions}
\end{aligned}$$

In this setting, $\phi(\mathbf{x}, \mathbf{w})$ is a multilayer perceptron that maps input data $\mathbf{x}$ to output y by using the set of weights w, and the set of parameters is expressed as $\theta \coloneqq (\mathbf{w}, \alpha, \tau)$. The model exhibits a posterior of dimension $d = 653$ and was applied to a 100-row dataset subsampled from the wine-red dataset.

We approximated the posterior of this model by using a variational diagonal Gaussian distribution, and we used the learning rate scheduler η as the step-decay function. The initial learning rate ($\in [10^{-5}, 10^{-2}]$ (MC,RQMC) or $\in [10^{-8}, 10^{-5}]$ (MLMC)) and the hyperparameters of the learning rate scheduler, β ($\in [0.1, 1]$) and r ($\in [10, 500]$), were optimized through the Tree-structured Parzen Estimator (TPE) sampler in optuna (Akiba et al., 2019) with 50 trials, where β and r are the decay and drop-rate parameter, respectively.

To optimize the variational free energy, we used the Adam optimizer for the MC- and RQMC-based methods and the SGD optimizer with a learning rate scheduler η for the MLMC-based method. We used 50 initial MC or RQMC samples for gradient estimation. We compared the empirical variance and SNR of these methods by using 50 or 10 initial MC or RQMC samples for inference.

	HBR			BLR			BNN		
	α_0	β	r	α_0	β	r	α_0	β	r
MC	0.39893	-	-	0.004735	-	-	0.007780	-	-
RQMC	0.39893	-	-	0.007780	-	-	0.007780	-	-
MLMC	0.027026	0.862527	221	0.007438	0.226316	458	9.062263e-6	0.819243	253

Table 3: Selected parameters optimized by optuna

Appendix G. Additional Experimental Results for Sensitivity of Optimization for Various Hyperparameter Settings

In this section, we conduct additional experiments to analyze our method's sensitivity for various hyperparameter settings and initial learning rates. As we mentioned in Section 4.1 regarding several theoretical analyses, the optimization performance of our method depends on the learning rate scheduler function η. To understand the characteristic of our method in more detail, we conducted benchmark experiments with various hyperparameter settings and initial learning rates to see how the performance of the optimization changes and compare it with the optimal settings in Section 5.1. The optimal hyperparameter settings selected by optuna are summarized in Table 3. The results are shown in Figures 8, 9, and 10.

These experimental results show that the optimization performance of our method significantly degrades when the drop-rate r and the initial learning rate are improperly set. This degradation reflects the property shown in Theorem 3, which shows that the learning rate scheduler function η affects the convergence of our method. Therefore, in practice, these hyperparameters should be carefully optimized according to the task and dataset in our method. A hyperparameter optimization library such as optuna (Akiba et al., 2019), which we used many times in this study, might be useful for this purpose.

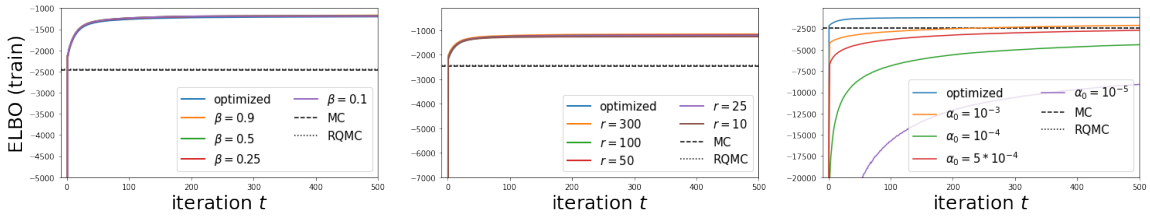

Figure 8: Experimental results for hierarchical linear regression analysis of various hyperparameter settings and initial learning rates. To confirm the optimization performance, the training ELBOs (higher is better) are lined up from the left.

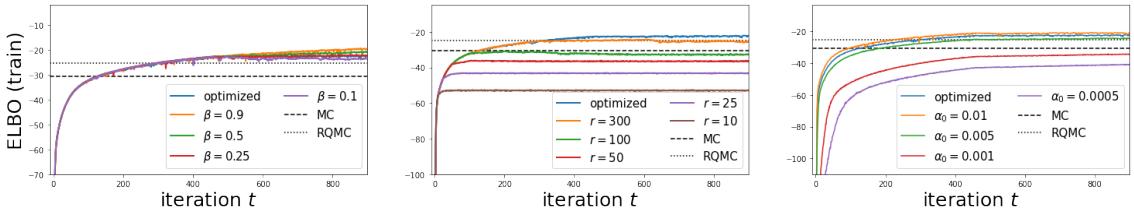

Figure 9: Experimental results for Bayesian logistic regression analysis of various hyperparameter settings and initial learning rates. To confirm the optimization performance, the training ELBOs (higher is better) are lined up from the left.

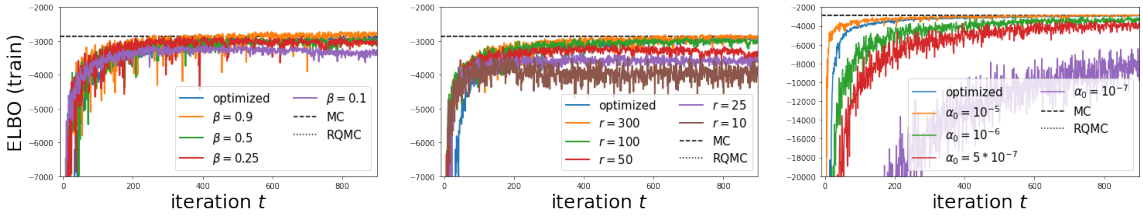

Figure 10: Experimental results for Bayesian neural network regression analysis of various hyperparameter settings and initial learning rates. To confirm the optimization performance, the training ELBOs (higher is better) are lined up from the left.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2623—2631, 2019.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 531–540, 2018.

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.

Léon Bottou, Frank E. Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. *arXiv preprint arXiv:1606.04838*, 2016.

Alexander Buchholz, Florian Wenzel, and Stephan Mandt. Quasi-Monte Carlo variational inference. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 668–677, 2018.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. In *International Conference on Learning Representations (ICLR)*, 2016.

K Andrew Cliffe, Michael B. Giles, Robert Scheichl, and Aretha L. Teckentrup. Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients. *Computing and Visualization in Science*, 14(1):3–15, 2011.

Justin Domke. Provable gradient variance guarantees for black-box variational inference. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, 2019.

Tomas Geffner and Justin Domke. Approximation based variance reduction for reparameterization gradients. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, 2020.

Anastasis Georgoulas, Jane Hillston, and Guido Sanguinetti. Unbiased Bayesian inference for population Markov jump processes via random truncations. *Statistics and Computing*, 27(4):991–1002, 2017.

Thomas Gerstner and Marco Noll. Randomized multilevel quasi-Monte Carlo path simulation. *Recent Developments in Computational Finance*, pages 349–369, 2013.

Michael B. Giles. Multi-level Monte Carlo path simulation. *Operations Research*, 56(3): 607–617, 2008.

Michael B. Giles. Multilevel Monte Carlo methods. *Monte Carlo and Quasi-Monte Carlo Methods 2012*, 56(3):79–98, 2013.

Michael B. Giles. Multilevel Monte Carlo methods. *Acta Numerica*, 24:259–328, 2015.

Michael B. Giles and Ben J. Waterhouse. Multilevel quasi-Monte Carlo path simulation. In *Advanced Financial Modelling, Radon Series on Computational and Applied Mathematics*, pages 165–181, 2009.

Michael B. Giles, Mateusz B. Majka, Lukasz Szpruch, Sebastian J. Vollmer, and Konstantinos C. Zygalakis. Multi-level Monte Carlo methods for the approximation of invariant measures of stochastic differential equations. *Statistics and Computing*, 30:507–524, 2020.

Mike Giles, Tigran Nagapetyan, Lukasz Szpruch, Sebastian Vollmer, and Konstantinos Zygalakis. Multilevel Monte Carlo for scalable Bayesian computations. *arXiv preprint arXiv:1609.06144*, 2016.

Paul Glasserman. Monte Carlo methods in financial engineering. *Stochastic Modelling and Applied Probability*, 53(1):XIII,596, 2003.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

Will Grathwohl, Dami Choi, Yuhuai Wu, Geoff Roeder, and David Duvenaud. Backpropagation through the Void: Optimizing control variates for black-box gradient estimation. In *International Conference on Learning Representations (ICLR)*, 2018.

Stefan Heinrich. Multilevel Monte Carlo methods. In *Proceedings of the Third International Conference on Large-Scale Scientific Computing-Revised Papers*, pages 58–67, 2001.

Philipp Hennig. Fast probabilistic optimization from noisy gradients. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pages 62–70, 2013.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations (ICLR)*, 2017.

Ajay Jasra, Seongil Jo, David Nott, Christine Shoemaker, and Raul Tempone. Multilevel Monte Carlo in approximate Bayesian computation. *arXiv preprint arXiv:1702.03628*, 2017.

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.

Sekitoshi Kanai, Yasuhiro Fujiwara, and Sotetsu Iwamura. Preventing gradient explosions in gated recurrent units. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 435–444, 2017.

Diederick P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

Pierre L'Ecuyer and Christiane Lemieux. Recent advances in randomized quasi-Monte Carlo methods. *Modeling Uncertainty*, pages 419–474, 2005.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. Directional analysis of stochastic gradient descent via von Mises-Fisher distributions in deep learning. *arXiv preprint arXiv:1810.00150*, 2018.

Christian Lemieux. *Monte Carlo and Quasi-Monte Carlo Sampling*. Springer, 2009.

Gunther Leobacher and Friedrich Pillichshammer. *Introduction to Quasi-Monte Carlo Integration and Applications*. Compact Textbooks in Mathematics. Springer, 2014.

Ximing Li, Changchun Li, Jinjin Chi, and Jihong Ouyang. Variance reduction in black-box variational inference by adaptive importance sampling. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJICAI)*, pages 2404–2410, 2018.

Zhiyuan Li and Sanjeev Arora. An exponential learning rate schedule for deep learning. In *International Conference on Learning Representations (ICLR)*, 2020.

Anne-Marie Lyne, Mark Girolami, Yves Atchadé, Heiko Strathmann, and Daniel Simpson. On Russian roulette estimates for Bayesian inference with doubly-intractable likelihoods. *Statistical Science*, 30(4):443–467, 2015.

Tomáš Mikolov. *Statistical language models based on neural networks*. PhD thesis, Brno University of Technology, 2012.

Andrew Miller, Nick Foti, Alexander D' Amour, and Ryan P Adams. Reducing reparameterization gradient variance. In *Advances in Neural Information Processing Systems 30 (NeurIPS)*, pages 3708–3718, 2017.

William J. Morokoff and Russel E. Caflisch. Quasi-random sequences and their discrepancies. *SIAM Journal on Scientific Computing*, 15:1251–1279, 1994.

Yadong Mu, Wei Liu, Xiaobai Liu, and Wei Fan. Stochastic gradient made stable: A manifold propagation approach for large-scale optimization. *IEEE Transactions on Knowledge & Data Engineering*, 29(02):458–471, 2017.

Shinichi Nakajima, Ryota Tomioka, Masashi Sugiyama, and S. Derin Babacan. Condition for perfect dimensionality recovery by variational Bayesian PCA. *Journal of Machine Learning Research*, 16(114):3757–3811, 2015.

Yurii E. Nesterov. A method for solving the convex programming problem with convergence rate $\mathcal{O}(1/k^2)$. *Soviet Mathematics Doklady*, 27:372–376, 1983.

Harald Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. SIAM, 1992.

John Paisley, David M. Blei, and Michael I. Jordan. Variational Bayesian inference with stochastic search. In *Proceedings of the 29th International Coference on International Conference on Machine Learning (ICML)*, pages 1363–1370, 2012.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pages 1310–1318, 2013.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*, pages 8024–8035, 2019.

Tom Rainforth, Adam R. Kosiorek, Tuan Anh Le, Chris J. Maddison, Maximilian Igl, Frank Wood, and Yee Whye Teh. Tighter variational bounds are not necessarily better. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 4274–4282, 2018.

Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 814–822, 2014.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 1278–1286, 2014.

Chang-han Rhee and Peter W. Glynn. Unbiased estimation with square root convergence for SDE models. *Operations Research*, 63(5):1026–1043, 2015.

Geoffrey Roeder, Yuhuai Wu, and David Duvenaud. Sticking the landing: Simple, lower-variance gradient estimators for variational inference. In *Advances in Neural Information Processing Systems 30 (NeurIPS)*, 2017.

Francisco J. R. Ruiz, Michalis K. Titsias, and David M. Blei. Overdispersed black-box variational inference. In *Uncertainty in Artificial Intelligence (UAI)*, 2016a.

Francisco J. R. Ruiz, Michalis K. Titsias, and David M. Blei. The generalized reparameterization gradient. In *Advances in Neural Information Processing Systems 29 (NeurIPS)*, 2016b.

Joseph Sakaya and Arto Klami. Importance sampled stochastic optimization for variational inference. In *Uncertainty in Artificial Intelligence (UAI)*, 2017.

Shai Shalev-Shwartz, Ohad Shamir, and Shaked Shammah. Failures of gradient-based deep learning. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 3067–3075, 2017.

Michalis K. Titsias and Miguel Lázaro-Gredilla. Doubly stochastic variational Bayes for non-conjugate inference. In *Proceedings of the 31th International Conference on Machine Learning (ICML)*, pages 1971–1979, 2014.

Michalis K. Titsias and Miguel Lázaro-Gredilla. Local expectation gradients for black box variational inference. In *Advances in Neural Information Processing Systems 28 (NeurIPS)*, pages 2638–2646, 2015.

Seiya Tokui and Issei Sato. Reparameterization trick for discrete variables. *arXiv preprint*, arXiv:1611.01239, 2016.

Minh-Ngoc Tran, David J. Nott, and Robert Kohn. Variational Bayes with intractable likelihood. *Journal of Computational and Graphical Statistics*, 26(4):873–882, 2017.

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 *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, pages 2627–2636, 2017.

George Tucker, Dieterich Lawson, Shixiang Gu, and Christopher Maddison. Doubly reparameterized gradient estimators for Monte Carlo objectives. In *International Conference on Learning Representations (ICLR)*, 2019.

David J. Warne, Ruth E. Baker, and Matthew J. Simpson. Multilevel rejection sampling for approximate Bayesian computation. *Computational Statistics & Data Analysis*, 124:71–86, 2018.

Ming Xu, Matias Quiroz, Robert Kohn, and Scott A. Sisson. Variance reduction properties of the reparameterization trick. In *Proceedings of the 22th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

Cheng Zhang, Judith Bütepage, Hedvig Kjellström, and Stephan Mandt. Advances in variational inference. *arXiv preprint*, arXiv:1711.05597, 2017.

Jingzhao Zhang, Tianxing He, Suvrit Sra, and Ali Jadbabaie. Why gradient clipping accelerates training: A theoretical justification for adaptivity. In *International Conference on Learning Representations (ICLR)*, 2020.