Wasserstein Gradient Flow over Variational Parameter Space for Variational Inference

Dai Hai Nguyen
Hokkaido University

Tetsuya Sakurai
University of Tsukuba

Hiroshi Mamitsuka
Kyoto University

Abstract

Variational Inference (VI) optimizes variational parameters to closely align a variational distribution with the true posterior, being approached through vanilla gradient descent in black-box VI or natural-gradient descent in natural-gradient VI. In this work, we reframe VI as the optimization of an objective that concerns probability distributions defined over a *variational parameter space*. Subsequently, we propose Wasserstein gradient descent for solving this optimization, where black-box VI and natural-gradient VI can be interpreted as special cases of the proposed Wasserstein gradient descent. To enhance the efficiency of optimization, we develop practical methods for numerically solving the discrete gradient flows. We validate the effectiveness of the proposed methods through experiments on synthetic and real-world datasets, supplemented by theoretical analyses.

1 Introduction

Many machine learning problems involve the challenge of approximating an intractable target distribution, which might only be known up to a normalization constant. Bayesian inference is a typical example, where the intractable and unnormalized target distribution is a result of the product of the prior and likelihood functions (Lindley, 1972; Von Toussaint, 2011). Variational Inference (VI), a widely employed approach across various application domains, seeks to approximate this intractable target distribution by utilizing a variational distribution (Blei et al., 2017; Jordan et al., 1999). VI is typically formulated as an optimization problem, with the objective of maximizing the evidence lower bound objective (ELBO), which is equivalent to minimizing the Kullback-Leiber (KL) divergence between the variational and target distributions.

The conventional method for maximizing the ELBO involves the use of gradient descent, such as black-box VI (BBVI) (Ranganath et al., 2014). The gradient of the ELBO can be expressed as an expectation over the variational distribution, which is typically estimated by Monte Carlo samples from this distribution. Natural-gradient-based methods, such as natural-gradient VI (NGVI) (Khan and Nielsen, 2018) has demonstrated its superior efficiency compared to the standard gradient descent for VI. The natural-gradient (Amari and Douglas, 1998) can be obtained from the vanilla gradient by preconditioning it with the inverse of the Fisher information matrix (FIM). However, explicitly computing this inverse FIM is expensive in general cases. An interesting fact highlighted by Khan and Nielsen (2018) is that the natural-gradient concerning the natural parameters of an exponential family distribution (e.g., Gaussian) is equivalent to the standard gradient concerning the expectation parameters. This equivalence simplifies the updates and often leads to faster convergence compared to gradient-based methods. Nevertheless, the natural-gradient methods generally do not accept simple updates when dealing with mixture models such as a Gaussian mixture. To overcome this problem, Lin et al. (2019); Gunawan et al. (2024) extend NGVI to mixture models which are more appropriate for complex and multi-modal posterior distributions.

Our work is motivated by the question: how can we extend gradient-based optimization methods for VI, such as BBVI and NGVI, to the cases where the variational distribution is a mixture of distributions (e.g., a Gaussian mixture)? Unlike the aforementioned methods that directly optimize for the variational parameters in VI, our approach imposes a mixing distribution over the variational parameters and optimizes this distribution using Wasserstein gradient flows (WGFs) (Jordan et al., 1998). In our approach, we can reframe VI as the optimization of an objective function related to

Proceedings of the 28[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

the mixing distribution. Then, we propose a preconditioned WGF over the space of variational parameters, using any quadratic form as the distance matrix, which can be a user-defined preconditioning matrix. Prior to our work, mixture models were handled by Wasserstein variational inference (WVI), which defines a WGF over the space of means and covariance matrices of Gaussian distributions, endowed with the Bures-Wasserstein distance (Bhatia et al., 2019).

In summary, our approach offers the following advantages:

First, we provide a *unified perspective on BBVI and NGVI*, showing that both updates can be precisely derived as particle approximation of our proposed WGFs over the variational parameter space (as shown in (18)), particularly when the number of particles is set to one. Leveraging well-established theories of WGF, we can *establish theoretical insights* into the proposed methods, which can deepen our understanding of behaviors of WGFs for VI. Additionally, by using multiple particles, each representing the variational parameters of a component, we can extend BBVI and NGVI to cases where the variational distribution is a *mixture of distributions*, which allow to improve the approximation of complex and multi-modal posterior distributions.

Second, our approach offers *more flexibility* than WVI due to the specification of variational parameters and preconditioning matrices to induce more efficient gradient flows in the variational parameter space. Specifically, we introduce two methods, GFlowVI and NGFlowVI, which perform better than WVI in experiments on both synthetic and real-world datasets. Furthermore, we also propose an *update formula for component weights* using mirror descent in the probability space with its theoretical analysis.

2 Related Work

We first review BBVI and NGVI, two commonly used gradient-based optimization methods for VI. Then, we provide background information on gradient flows on the probability distribution space. In addition, we present some relevant hierarchical variational models and highlight the key distinctions between them and our approach.

2.1 Gradient-based Optimization for VI

We consider the following problem setting. Let D be a set of observations, $\mathbf{z}$ be a latent variable and $q(\mathbf{z}|\boldsymbol{\lambda})$ be the variational distribution with the variational parameter $\boldsymbol{\lambda} \in \mathbb{R}^d$, our goal is to approximate the true posterior $\pi(\mathbf{z}|D)$ with q by minimizing the negated ELBO:

$$\min_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{\lambda}) = E_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})}\left[f(\mathbf{z})\right] - H(q), \tag{1}$$

where $f(\mathbf{z}) = -\log \pi(D, \mathbf{z})$ and $H(q)$ is the entropy of q, given by: $H(q) = -\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})}\left[\log q(\mathbf{z}|\boldsymbol{\lambda})\right]$.

The negated ELBO can be optimized with the gradient descent algorithm, known as BBVI (Ranganath et al., 2014). To estimate the gradient of negated ELBO, we can use the reparameterization trick (Kingma and Welling, 2013), which reparameterizes $q(\mathbf{z}|\boldsymbol{\lambda})$ in terms of a surrogate random variable $\epsilon \sim p(\epsilon)$ and a deterministic function $g_{\boldsymbol{\lambda}}$ in such a way that sampling from $q(\mathbf{z}|\boldsymbol{\lambda})$ is performed as follows: $\epsilon \sim p(\epsilon)$, $\mathbf{z} = g_{\boldsymbol{\lambda}}(\epsilon)$. If $g_{\boldsymbol{\lambda}}$ and p are continuous with respect to $\mathbf{z}$ and ϵ, respectively, the gradient of negated ELBO and parameter update are as follows:

$$\begin{aligned} \boldsymbol{\lambda}_{n+1} \leftarrow& \boldsymbol{\lambda}_n - \eta \nabla_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{\lambda}_n), \\ \nabla_{\boldsymbol{\lambda}} \mathcal{L}(\boldsymbol{\lambda}) =& \mathbb{E}_{\epsilon\sim p}\left[\nabla_{\boldsymbol{\lambda}}\left(f(g_{\boldsymbol{\lambda}}(\epsilon)) + \log q(g_{\boldsymbol{\lambda}}(\epsilon)|\boldsymbol{\lambda})\right)\right], \end{aligned} \tag{2}$$

where η is the learning rate, and $\nabla_{\boldsymbol{\lambda}}\mathcal{L}$ can be estimated using Monte Carlo samples from $p(\epsilon)$.

Compared to the gradient descent, natural-gradient descent has been shown to be much more efficient for VI (Khan and Nielsen, 2018). The natural-gradient descent can be obtained from the standard gradient descent by preconditioning it with the inverse Fisher Information Matrix (FIM), as follows: $\boldsymbol{\lambda}_{n+1} \leftarrow \boldsymbol{\lambda}_n - \eta\left[\mathbf{F}(\boldsymbol{\lambda}_n)\right]^{-1}\nabla_{\boldsymbol{\lambda}}\mathcal{L}(\boldsymbol{\lambda}_n)$, where $\mathbf{F}(\boldsymbol{\lambda})$ is the FIM with respect to $\boldsymbol{\lambda}$. However, explicitly computing the FIM can be expensive. As a more efficient alternative, Khan and Nielsen (2018) show that when q is an exponential family distribution and $\boldsymbol{\lambda}$ is its natural parameter, the natural-gradient with respect to $\boldsymbol{\lambda}$ is equivalent to the standard gradient with respect to the expectation parameter:

$$\boldsymbol{\lambda}_{n+1} \leftarrow \boldsymbol{\lambda}_n - \eta \nabla_{\mathbf{m}} \mathcal{L}(\boldsymbol{\lambda}_n), \tag{3}$$

where $\mathbf{m}$ is the expectation parameter of q, given by: $\mathbf{m}(\boldsymbol{\lambda}) = \mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})}\left[T(\mathbf{z})\right]$, where $T(\mathbf{z})$ is the sufficient statistics of q (Blei et al., 2017). For many existing works on natural-gradient for VI, e.g., Khan and Nielsen (2018), the above gradient is easier to compute than the gradient with respect to $\boldsymbol{\lambda}$ and the natural-gradient descent admits a simpler update form than gradient descent. For instance, when q is a diagonal Gaussian, i.e., $q(\mathbf{z}|\boldsymbol{\lambda}) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$, the update (3) becomes:

$$\begin{aligned} \boldsymbol{\mu}_{n+1} \leftarrow& \boldsymbol{\mu}_n - \eta \boldsymbol{\sigma}^2_{n+1} \odot \nabla_{\mathbf{z}} f(\mathbf{z}), \\ \boldsymbol{\sigma}^{-2}_{n+1} \leftarrow& (1-\eta)\boldsymbol{\sigma}_n^{-2} + \eta \texttt{diag}\left[\nabla^2_{\mathbf{z}} f(\mathbf{z})\right], \end{aligned} \tag{4}$$

where $\mathbf{a} \odot \mathbf{b}$ denotes the element-wise product between vectors $\mathbf{a}$ and $\mathbf{b}$ and $\texttt{diag}[\mathbf{A}]$ denotes the function to extract diagonal entries of matrix $\mathbf{A}$.

2.2 Gradient Flows on Probability Distribution Space

Consider the problem of minimizing $F : \mathcal{P}(\Omega) \to \mathbb{R}$, a functional in the space of probability distributions $\mathcal{P}(\Omega)$ with $\Omega \subset \mathbb{R}^d$. We first endow a Riemannian geometry on $\mathcal{P}(\Omega)$, characterized by the second-order Wasserstein (or 2-Wasserstein) distance between two distributions: $\mathcal{W}_2^2(\rho, \rho') = \inf_\gamma \left\{ \int_{\Omega\times\Omega} \|\mathbf{x} - \mathbf{x}'\|_2^2 \mathrm{d}\gamma(\mathbf{x}, \mathbf{x}') : \gamma \in \Gamma(\rho, \rho') \right\}$, where $\Gamma(\rho, \rho')$ is the set of all possible couplings with marginals ρ and ρ'. This is an optimal transport problem, which has been shown effective for comparing probability distributions in many applications, such as (Petric Maretic et al., 2019; Nguyen et al., 2021; Nguyen and Tsuda, 2023; Nguyen and Sakurai, 2024; Caluya and Halder, 2019). If ρ is absolutely continuous with respect to the Lebesgue measure, there exists a unique optimal transport plan from ρ to ρ', i.e., a mapping $T : \Omega \to \Omega$ pushing ρ onto ρ' satisfying $\rho' = T\#\rho$, where $T\#p$ denotes the pushforward measure of ρ. Then, the 2-Wasserstein distance can be equivalently reformulated as: $\mathcal{W}_2^2(\rho, \rho') = \inf_T \int_\Omega \|\mathbf{x} - T(\mathbf{x})\|_2^2 \mathrm{d}\rho(\mathbf{x})$.

Let $\{\rho_t\}_{t\in[0,1]}$ be an absolutely continuous curve in $\mathcal{P}(\Omega)$ with finite second-order moments. Then, for $t \in [0, 1]$, there exists a velocity field $v_t \in L^2(\rho_t)$, where $L^2(\rho_t)$ denotes the space of function $h : \Omega \to \Omega$, such that the *continuity equation* $\partial_t \rho_t + \mathrm{div}(\rho_t v_t) = 0$ is satisfied, where div is the divergence operator (Santambrogio, 2015). Consider two distribution ρ_t and ρ_{t+h}, and let T_h be the optimal transport map between them. Define $v_t(\mathbf{x})$ as the discrete velocity of the particle $\mathbf{x}$ at time t, given by $v_t(\mathbf{x}) = (T_h(\mathbf{x}) - \mathbf{x})/h$ (i.e., displacement/time). It is shown that, in the limit $h \to 0$, v_t has the following form: $v_t(\mathbf{x}) = -\nabla \delta F(\rho_t)(\mathbf{x})$, where $\delta F(\rho_t)$ is the first variation of F at ρ_t. By the continuity equation, we get the expression of WGF of F, as follows:

$$\partial_t \rho_t = \mathrm{div}\left(\rho_t \nabla \delta F(\rho_t)\right). \tag{5}$$

Particle-based variational inference (ParVI) methods (Liu and Wang, 2016; Liu et al., 2019) use a set of particles $\{\mathbf{x}_{k,t}\}_{k=1}^K$ to approximate ρ_t and update the particles to approximate the WGF. Each particle $\mathbf{x}_{k,t}$ is then updated as follows:

$$d\mathbf{x}_{k,t} = \tilde{v}_t(\mathbf{x}_{k,t})dt \tag{6}$$

where $\tilde{v}_t$ is an approximation of v_t obtained from the empirical distribution $\tilde{\rho}_t$. Therefore, different ParVI methods can be derived by selecting appropriate $\tilde{v}_t$ and discretizing (6) using specific schemes such as the first order explicit Euler discretization (Liu et al., 2019).

2.3 Hierarchical Variational Models

Several methods are related to our approach, including SIVI (Yin and Zhou, 2018), SIVI-SM (Yu and Zhang, 2023), Particle SIVI (Lim and Johansen, 2025), SMI (Rønning et al., 2024), WVI (Lambert et al., 2022). In this subsection, we highlight the distinctions between these methods and ours.

SIVI, SIVI-SM, Particle SIVI, SMI and our methods are based on hierarchical variational models, where the variational distribution is defined as a mixture model. Both SIVI (Yin and Zhou, 2018) and our methods define q as a mixture and optimize the mixing distribution, $\rho(\boldsymbol{\lambda})$, rather than q directly. This strategy mitigates the limitations of traditional variational families. However, due to the intractability of the variational distributions' densities, SIVI either uses a surrogate ELBO (a lower bound of ELBO) or relies on costly inner-loop MCMC runs for ELBO maximization during training. To address these challenges, SIVI-SM (Yu and Zhang, 2023) introduces score matching for training. In contrast, our methods optimize the ELBO directly using preconditioned Wasserstein gradient descent (WGD) in the variational parameter space.

Furthermore, our methods are closely related to Particle SIVI (Lim and Johansen, 2025) and SMI (Rønning et al., 2024), both of which represent the variational distributions with particles, where each particle corresponds to the parameters of a component (e.g., mean and variance of a Gaussian distribution). This particle representation enhances the expressiveness of the variational distributions, enabling them to more effectively approximate complex targets compared to other particle-based variational inference methods, such as Stein variational gradient descent (SVGD) (Liu and Wang, 2016). However, our methods build on the well-established theory of WGD, offering a unified perspective on BBVI and NGVI, while also providing theoretical insights into these methods for VI.

Pior to our work, VI was handled by gradient flows defined in the Brures-Wasserstein space (Lambert et al., 2022; Yi and Liu, 2023), a subspace of the Wasserstein space consisting of Gaussian distributions. Compared to this approach, ours offers greater flexibility due to the specification of variational parameters and preconditioning matrices, which induces more efficient gradient flows in the variational parameter space.

Finally, we introduce a new update rule for component weights using mirror descent in the probability space, accompanied by its theoretical analysis (see Subsection 3.2). This aspect is not addressed by the previous methods, including Yin and Zhou (2018); Yu and Zhang (2023); Lim and Johansen (2025); Rønning et al. (2024); Lambert et al. (2022); Yi and Liu (2023).

3 Proposed Methods

3.1 Gradient Flows over Variational Parameter Space

Our perspective is motivated from the key question: how to extend gradient-based optimization, such as BBVI and NGVI, to the case where the variational distribution is assumed to be a mixture of distributions (such as a Gaussian mixture). Following the key observation already made by Chen et al. (2018), we identify the variational distribution with a distribution over the variational parameters. Specifically, the variational distribution q now corresponds to a mixture of an infinite number of components as follows: $q(\mathbf{z}) = \int_\Omega q(\mathbf{z}|\boldsymbol{\lambda})d\rho(\boldsymbol{\lambda})$, where Ω denotes the variational parameter space and ρ is the probability distribution over Ω. As a result, we can reformulate the negated ELBO with respect to variational parameters into a distributional optimization problem with respect to ρ over variational parameters as follows:

$$\min_{\rho\in\mathcal{P}(\Omega)} \mathcal{L}(\rho) = \mathbb{E}_{\boldsymbol{\lambda}\sim\rho}\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})}\left[f(\mathbf{z}) + \log q(\mathbf{z})\right], \quad (7)$$

where $\mathcal{P}(\Omega)$ denotes the set of distributions over variational parameters in the context of our work.

Remark 1. It is noteworthy that both VI, as expressed in (1) and our reformulated problem, as expressed in (7), involve the optimization over probability measure spaces. However, the fundamental distinction lies in the definitions of the domain Ω: in (2), the optimization variable is the variational distribution $q(\mathbf{z}|\boldsymbol{\lambda})$, which is defined within the domain of the *latent variable* $\mathbf{z}$, while in (7), the variable is ρ, which is defined within the spaces of *variational parameter* $\boldsymbol{\lambda}$.

The following theorem shows the first variation of $\mathcal{L}(\rho)$. This is particularly useful in formulating the gradient flows on the probability distribution space of variational parameters.

Theorem 1. *(First variation of $\mathcal{L}(\rho)$). The first variation of $\mathcal{L}(\rho)$ defined in (7) is given by:*

$$\delta\mathcal{L}(\rho)(\boldsymbol{\lambda}) = \mathbb{E}_{\boldsymbol{z}\sim q(\cdot|\boldsymbol{\lambda})}\left[f(\boldsymbol{z}) + \log\left(\mathbb{E}_{\boldsymbol{\lambda}'\sim\rho}\left[q(\boldsymbol{z}|\boldsymbol{\lambda}')\right]\right)\right] + 1, \quad (8)$$

which can be approximated using Monte Carlo samples:

$$\delta\mathcal{L}(\rho)(\boldsymbol{\lambda}) \approx \frac{1}{S}\sum_{i=1}^{S}\left[f(\boldsymbol{z}_i) + \log\left(\frac{1}{K}\sum_{k=1}^{K} q(\boldsymbol{z}_i|\boldsymbol{\lambda}_k)\right)\right] + 1,$$

where $\boldsymbol{\lambda}_k \sim \rho$, $k = 1, 2, ..., K$ and $\boldsymbol{z}_i \sim q(\cdot|\boldsymbol{\lambda})$, $i = 1, ..., S$.

The proof of Theorem 1 can be found in Appendix A. Our objective is to establish gradient flows over probability distribution spaces, where the domain Ω is defined over variational parameters. The Wasserstein gradient flow is essentially a curve $\{\rho_t\}_{t\in[0,1]}$ that satisfies (5). In this work, we consider a preconditioned gradient flow as follows:

$$\frac{\partial\boldsymbol{\lambda}_t}{\partial t} = -\mathbf{C}(\boldsymbol{\lambda}_t)\nabla_{\boldsymbol{\lambda}}\delta\mathcal{L}(\rho_t)(\boldsymbol{\lambda}_t), \quad (9)$$

where $\mathbf{C}(\boldsymbol{\lambda}) \in \mathbb{R}^{d\times d}$ is a positive-definite *preconditioning matrix*. Then, the dynamic of ρ_t, the probability distribution of $\boldsymbol{\lambda}_t$, is induced by the following continuity equation:

$$\frac{\partial\rho_t}{\partial t} + \mathtt{div}(\rho_t\mathbf{C}v_t) = 0, v_t = -\nabla_{\boldsymbol{\lambda}}\delta\mathcal{L}(\rho_t). \quad (10)$$

Continuous-time dynamics. We study the dissipation of $\mathcal{L}(\rho_t)$ along the trajectory of the flow (10), as stated in the following proposition.

Proposition 2. *The dissipation of $\mathcal{L}$ along the gradient flow (10) is characterized as follows:*

$$\frac{d\mathcal{L}(\rho_t)}{dt} = -\langle v_t, \boldsymbol{C}v_t\rangle_{L^2(\rho_t)}, \quad (11)$$

where $\langle\cdot,\cdot\rangle_{L^2(\rho)}$ denotes the inner product of $L^2(\rho)$.

The proof of Proposition 2 can be found in Appendix B. Since $\mathbf{C}$ is a positive-definite matrix, the right-hand side of (11) is non-positive. Thus Proposition 2 indicates that $\mathcal{L}$ with respect to ρ_t decreases along the gradient flow (10). The second consequence is the following corollary.

Corollary. *For any $t > 0$, we have:*

$$\min_{0\le s\le t}\langle v_s, \boldsymbol{C}v_s\rangle_{L^2(\rho_t)} \le \frac{1}{t}\int_0^t \langle v_s, \boldsymbol{C}v_s\rangle_{L^2(\rho_s)}ds \le \frac{\mathcal{L}(\rho_0) - \min_{\rho\in\mathcal{P}(\Omega)}\mathcal{L}(\rho)}{t}.$$

The corollary indicates that the gradient norm will converge to zero as t goes to infinite. However, it is not guaranteed that it converges to the globally optimal solution because of the non-convexity of $\mathcal{L}$.

Discrete-time dynamics. Next we study the dissipation of $\mathcal{L}$ in discrete time. We consider the following gradient descent update in the Wasserstein space applied to $\mathcal{L}$ at each iteration $n \ge 0$:

$$\rho_{n+1} = (\mathbf{I} - \eta\mathbf{C}v_n)_{\#}\rho_n, \quad (12)$$

where $\mathbf{I}$ is the identity map. This update corresponds to a forward Euler discretization of the gradient flow (10). Let $\rho_0 \in \mathcal{P}(\Omega)$ be the initial distribution of parameter $\boldsymbol{\lambda}_0$, i.e. $\boldsymbol{\lambda}_0 \sim \rho_0$. For every $n > 0$, $\boldsymbol{\lambda}_n \sim \rho_n$, we have:

$$\boldsymbol{\lambda}_{n+1} = \boldsymbol{\lambda}_n - \eta\mathbf{C}(\boldsymbol{\lambda}_n)v_n(\boldsymbol{\lambda}_n). \quad (13)$$

We study the dissipation of $\mathcal{L}(\rho_n)$ along the gradient update (13) in the infinite number of particles regimes (where K goes to infinity). We intend to obtain a descent lemma similar to Proposition 2. However, the discrete-time analysis requires more assumptions than the continuous-time analysis. Here we assume the following for all $\boldsymbol{\lambda}$:

(A1) Assume $\exists \alpha > 0$ s.t. $\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})} \|\nabla_{\boldsymbol{\lambda}} \log q(\mathbf{z}|\boldsymbol{\lambda})\|_2^2 \leq \alpha$.

(A2) Assume $\exists \beta > 0$ s.t. $\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})} \|\nabla^2_{\boldsymbol{\lambda}} \log q(\mathbf{z}|\boldsymbol{\lambda})\|_{\text{op}} \leq \beta$, where $\|\mathbf{A}\|_{\text{op}} = \sup_{\|\mathbf{u}\|_2=1} \|\mathbf{A}\mathbf{u}\|_2$ is the operator norm of matrix $\mathbf{A}$.

(A3) Assume $\exists M_1, M_2 > 0$ s.t. $\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})} |f(\mathbf{z})| \leq M_1, \mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})} |\log q(\mathbf{z})| \leq M_2$.

Assumptions **(A1)** and **(A2)** may not hold in general cases. For instance, when $q(\mathbf{z}|\boldsymbol{\lambda})$ is a Gaussian and $\boldsymbol{\lambda} = \left[\boldsymbol{\mu}, \boldsymbol{\Sigma}^{-1}\right]$, the gradient and Hessian of $\log q(\mathbf{z}|\boldsymbol{\lambda})$ with respect to $\boldsymbol{\Sigma}^{-1}$ cannot be bounded. However, it is possible to make **(A1)** and **(A2)** hold by imposing constraints on the covariance matrix $\boldsymbol{\Sigma}$. Suppose that $a\mathbf{I} \preceq \boldsymbol{\Sigma} \preceq b\mathbf{I}$ $(0 < a < b)$, it can be verified that by setting $\alpha_1 = \alpha_2 = 1/4b + a^{-2}b$ and $\beta_1 = \beta_2 = a^{-1}$, Assumptions **(A1)** and **(A2)** hold. We discuss how to tackle the constraints during the optimization process in Subsection 3.4.

Given our assumptions, we quantify the decreasing of $\mathcal{L}$ along the gradient update (13), as follows.

Proposition 3. *Assume **(A1)**,**(A2)** and **(A3)** hold. Let $\kappa = (\alpha + \beta)(M_1 + M_2)$, and choose sufficiently small learning rate $\eta < 2/\kappa$. Then we have:*

$$\mathcal{L}(\rho_{n+1}) - \mathcal{L}(\rho_n) \leq -\eta \left(1 - \kappa\frac{\eta}{2}\right) \langle v_n, \boldsymbol{C} v_n\rangle_{L^2(\rho_n)}. \tag{14}$$

The proof of Proposition 3 can be found in Appendix C. Proposition 3 indicates that the objective $\mathcal{L}(\rho_n)$ decreases by the gradient update (13) since the right-hand side of (14) is non-positive by choosing a sufficiently small learning rate and the positive-definiteness of $\mathbf{C}$. The following corollary is directly derived from the descent lemma.

Corollary. *Let $\eta < 2/\kappa$ and $c_\eta = \eta\left(1 - \kappa\frac{\eta}{2}\right)$. Then, we have:*

$$\begin{aligned}\min_{i=1,2,\ldots,n} \langle v_i, \boldsymbol{C} v_i\rangle_{L^2(\rho_i)} \leq& \frac{1}{n}\sum_{i=1}^{n} \langle v_i, \boldsymbol{C} v_i\rangle_{L^2(\rho_i)} \\ \leq& \frac{\mathcal{L}(\rho_0) - \min_{\rho\in\mathcal{P}(\Omega)} \mathcal{L}(\rho)}{c_\eta n}.\end{aligned}$$

The corollary indicates that the gradient norm will converge to zero as n increases. However, similar to the argument mentioned in the continuous-time analysis, it is not guaranteed that it converges to the globally optimal solution because of the non-convexity of $\mathcal{L}$.

3.2 Weight Update via Infinite-dimensional Mirror Gradient Iterates

In (13), only particle position, i.e. $\boldsymbol{\lambda}_n$, is updated, while its weight $\rho_n(\boldsymbol{\lambda}_n)$ is kept fixed throughout the optimization. This weight restriction may limit the approximation capacity of q, especially when the number of particles is limited. To address it, we propose a scheme to update the weights of particles via the infinite-dimensional Mirror Descent (MD).

Theorem 4. *(Infinite-dimensional MD) We define an iterate of infinite-dimensional Mirror Descent as follows: given $\mu \in \mathcal{P}(\Omega)$, the learning rate η, and a function $g : \Omega \to \mathbb{R}$, we have:*

$$\begin{aligned}\mu^+ =& MD_\eta(\mu, g) \\ =& \arg\min_{\rho\in\mathcal{P}(\Omega)} \left\{\eta \int_\Omega g(\boldsymbol{\lambda})(\rho(\boldsymbol{\lambda}) - \mu(\boldsymbol{\lambda}))d\boldsymbol{\lambda} + KL(\rho, \mu)\right\},\end{aligned} \tag{15}$$

which can be equivalently defined as follows: for all $\boldsymbol{\lambda} \in \Omega$, $\mu^+(\boldsymbol{\lambda}) \propto \mu(\boldsymbol{\lambda}) \exp\left(-\eta g(\boldsymbol{\lambda})\right)$.

The proof of Theorem 4 is straightforwardly extended from the celebrated entropy mirror descent in finite dimensional space (see Hsieh et al. (2019) for more details). With the MD iterate defined above, we can define the following updates for both particle positions and their weights:

$$\bar{\rho}_n = (\mathbf{I} - \eta\mathbf{C}\nabla_{\boldsymbol{\lambda}}\delta\mathcal{L}(\rho_n))_{\#}\rho_n, \rho_{n+1} = \text{MD}_\eta(\bar{\rho}_n, \delta\mathcal{L}(\bar{\rho}_n)), \tag{16}$$

where the first update of (16), corresponding to the Wasserstein transport, is responsible for updating the particles positions, i.e. $\boldsymbol{\lambda}$, while the second update, corresponding to the Mirror Descent part, is responsible for updating the weights, i.e. $\rho_n(\boldsymbol{\lambda})$. We show the following descent lemma for (16).

Proposition 5. *Assume **(A1)**,**(A2)** and **(A3)** hold. Let $\kappa = (\alpha + \beta)(M_1 + M_2)$, and choose sufficiently small learning rate $\eta < \min\{2/\kappa, 1\}$. Then we have:*

$$\begin{aligned}\mathcal{L}(\rho_{n+1}) - \mathcal{L}(\rho_n) \leq& -\eta\left(1 - \kappa\frac{\eta}{2}\right)\langle v_n, \boldsymbol{C} v_n\rangle_{L^2(\rho_n)} \\ &- \left(\frac{1}{\eta} - 1\right) KL(\rho_{n+1}, \bar{\rho}_n)\end{aligned} \tag{17}$$

The proof of Proposition 5 can be found in Appendix D. Compared to Proposition 3, Proposition 5 demonstrates a stronger decrease per iteration, attributed to the non-negative KL term, highlighting the advantage of

incorporating MD iterates to enhance the convergence of our proposed updates.

Remark 2. We emphasize that the proposed updates (16) are closely related to Wasserstein-Fisher-Rao gradient flow of $\mathcal{L}$ (Gallouët and Monsaingeon, 2017). Specifically, we demonstrate in Appendix E that the second update of (16) aligns with the Fisher-Rao gradient flow as η approaches 0. As a result, the proposed updates (16) can be viewed as the discrete approximation of the *preconditioned* version of Wasserstein-Fisher-Rao gradient flow of $\mathcal{L}$.

3.3 Particle Approximation of Gradient Flows

For solving problem (7) using the updates (16), we assume that ρ_n is described by a set of particles $\{\boldsymbol{\lambda}_{k,n}\}_{k=1}^K$ and weights $\{a_{k,n}\}_{k=1}^K$. Then, the variational distribution $q_n(\mathbf{z})$ at iteration n corresponds to a familiar mixture model with a finite number of components as follows: $q_n(\mathbf{z}) = \mathbb{E}_{\boldsymbol{\lambda}\sim\rho_n}\left[q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\right] = \sum_{k=1}^K a_{k,n} q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})$. We perform the first update of (16) on particle positions, as follows:

$$\boldsymbol{\lambda}_{k,n+1} = \boldsymbol{\lambda}_{k,n} - \eta \mathbf{C}(\boldsymbol{\lambda}_{k,n}) \nabla \delta\mathcal{L}(\rho_n)(\boldsymbol{\lambda}_{k,n}). \tag{18}$$

We perform the second update of (16) on the weights of particles as follows:

$$a_{k,n+1} \propto a_{k,n} \exp(-\eta \delta\mathcal{L}(\bar{\rho}_n)(\boldsymbol{\lambda}_{k,n+1})), \tag{19}$$

where $\bar{\rho}_n(\boldsymbol{\lambda}) = \sum_{k=1}^K a_{k,n}\delta_{\boldsymbol{\lambda}_{k,n+1}}(\boldsymbol{\lambda})$. We now demonstrate the simplicity of our update (18) when $q(\mathbf{z}|\boldsymbol{\lambda})$ is a diagonal Gaussian distribution.

Gradient flow VI (GFlowVI). First, we consider the case $\mathbf{C} = \mathbf{I}$. Let $\boldsymbol{\lambda}_{k,n} = (\boldsymbol{\mu}_{k,n}, \mathbf{s}_{k,n})$ be the k-th variational parameter at the n-th iteration, where $\mathbf{s}_{k,n} = \boldsymbol{\sigma}_{k,n}^{-2}$ for $k = 1, 2, ..., K$. Each variational parameter corresponds to a Gaussian distribution, and so we refer it to as a "Gaussian particle". For each iteration n, we generate a sample $\mathbf{z}$ from $q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})$. Then we update $\boldsymbol{\mu}_{k,n}$ and $\mathbf{s}_{k,n}$ as follows:

$$\begin{aligned}
&\boldsymbol{\mu}_{k,n+1} = \boldsymbol{\mu}_{k,n} - \eta\left[\nabla_{\mathbf{z}} f(\mathbf{z}) + \nabla_{\mathbf{z}} \log q_n(\mathbf{z})\right] \\
&- \eta \mathbf{w}_k \nabla_{\boldsymbol{\mu}_k} \log q(\mathbf{z}|\boldsymbol{\lambda}_{k,n}). \\
&\mathbf{s}_{k,n+1} = \mathbf{s}_{k,n} - \eta \mathbf{w}_k \nabla_{\mathbf{s}_k} \log q(\mathbf{z}|\boldsymbol{\lambda}_{k,n}) \\
&+ \frac{\eta}{2} \oslash (\mathbf{s}_{k,n} \odot \mathbf{s}_{k,n}) \odot \mathtt{diag}\left[\nabla_{\mathbf{z}}^2 f(\mathbf{z}) + \nabla_{\mathbf{z}}^2 \log q_n(\mathbf{z})\right]
\end{aligned} \tag{20}$$

where $\mathbf{w}_k = q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})/q_n(\mathbf{z})$, and $\mathbf{a} \oslash \mathbf{b}$ denotes the element-wise division between vectors $\mathbf{a}$ and $\mathbf{b}$. We refer to this update as gradient-flow VI (GFlowVI). The update (20) is derived using the Bonnet's and Price's theorems (Bonnet, 1964; Price, 1958) for the Gaussian distribution $q(\mathbf{z}|\boldsymbol{\lambda}) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$:

$$\begin{aligned}
\nabla_{\boldsymbol{\mu}} \mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})}\left[f(\mathbf{z})\right] &= \mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})}\left[\nabla_{\mathbf{z}} f(\mathbf{z})\right], \\
\nabla_{\boldsymbol{\Sigma}} \mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})}\left[f(\mathbf{z})\right] &= \frac{1}{2}\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})}\left[\nabla_{\mathbf{z}}^2 f(\mathbf{z})\right].
\end{aligned} \tag{21}$$

Natural-gradient flow VI (NGFlowVI). Next, we consider the case that $\mathbf{C} = \mathbf{F}^{-1}$ is the inverse FIM. As much discussed in previous studies (e.g., Khan and Nielsen (2018)), the natural-gradient update does not need inverting the **FIM** for specific types of models and applications, e.g. exponential-family distributions. Thus, in this case, we consider $\boldsymbol{\lambda}$ to be the natural parameters of the Gaussian $q(\mathbf{z}|\boldsymbol{\lambda})$. Specifically, the natural parameters and expectation parameters can be defined as follows:

$$\boldsymbol{\lambda}_{k,n}^{(1)} = \mathbf{s}_{k,n} \odot \boldsymbol{\mu}_{k,n}, \boldsymbol{\lambda}_{k,n}^{(2)} = -\frac{1}{2}\mathbf{s}_{k,n}$$

and

$$\mathbf{m}_{k,n}^{(1)} = \boldsymbol{\mu}_{k,n}, \mathbf{m}_{k,n}^{(2)} = \boldsymbol{\mu}_{k,n} \odot \boldsymbol{\mu}_{k,n} + 1 \oslash \mathbf{s}_{k,n}.$$

The computational efficiency of the natural-gradients is a result of the following relation:

$$\mathbf{F}^{-1}(\boldsymbol{\lambda}_{k,n}) \nabla_{\boldsymbol{\lambda}} \delta\mathcal{L}(\delta\rho_n)(\boldsymbol{\lambda}_{k,n}) = \nabla_{\mathbf{m}} \delta\mathcal{L}(\delta\rho_n)(\boldsymbol{\lambda}_{k,n}). \tag{22}$$

Using the relation (21) and relation in Khan and Nielsen (2018), the update (18) becomes:

$$\begin{aligned}
\boldsymbol{\mu}_{k,n+1} =& \boldsymbol{\mu}_{k,n} - \eta\left[\nabla_{\mathbf{z}} f(\mathbf{z}) + \nabla_{\mathbf{z}} \log q_n(\mathbf{z})\right] \oslash \mathbf{s}_{k,n+1} \\
&- \eta \mathbf{w}_k \nabla_{\boldsymbol{\mu}_k} \log q(\mathbf{z}|\boldsymbol{\lambda}_{k,n}) \oslash \mathbf{s}_{k,n+1} \\
\mathbf{s}_{k,n+1} =& \mathbf{s}_{k,n} + \eta\, \mathtt{diag}\left[\nabla_{\mathbf{z}}^2 f(\mathbf{z}) + \nabla_{\mathbf{z}}^2 \log q_n(\mathbf{z})\right] \\
&- 2\eta \mathbf{w}_k (\mathbf{s}_{k,n} \odot \mathbf{s}_{k,n}) \odot \nabla_{\mathbf{s}_k} \log q(\mathbf{z}|\lambda_{k,n})
\end{aligned} \tag{23}$$

We refer to this update as natural-gradient flow VI (NGFlowVI). Detailed derivations of the updates (20) and (23) can be found in Appendix F.

Remark 3. Note that the gradient update (18) can serve as a generalization of BBVI and NGVI. Indeed, when $K = 1$, $\mathbf{C} = \mathbf{I}$ and $\boldsymbol{\lambda}_n = (\boldsymbol{\mu}_n, \boldsymbol{\sigma}_n)$, the update (18) recovers the update (2) of BBVI with the reparameterization trick. Also, when $K = 1$, $\mathbf{C} = \mathbf{F}^{-1}$ and $\boldsymbol{\lambda}_n = \left(\boldsymbol{\sigma}_n^{-2} \odot \boldsymbol{\mu}_n, -1/2\boldsymbol{\sigma}_n^{-2}\right)$, the update (18) recovers the update (4) of NGVI.

3.4 A Simple Fix to Negative Hessian Problem

In the updates (20) and (23), the vectors $\mathbf{s}_{k,n}$ (for $k = 1, 2, ..., K$) are updated based on the Hessian of $f(\mathbf{z}) + \log q(\mathbf{z})$. Since $f(\mathbf{z}) + \log q(\mathbf{z})$ is a non-convex, its

Hessian may not be positive-definite, leading to instability. Although the generalized Gaussian-Newton approximation from Khan and Nielsen (2018) is suggested to address it, it does not effectively solve the problem when applied to $f(\mathbf{z})+\log q(\mathbf{z})$. We introduce a solution to this issue by the approach by Nguyen and Sakurai (2023), which updates particles within a constrained domain. We observe that in the updates (20) and (23), the variance vectors appear independently in the second part of the variational parameters, i.e. $\boldsymbol{\lambda}^{(2)}_{k,n} = \mathbf{s}_{k,n}$ for the first case ($\mathbf{C} = \mathbf{I}$) and $\boldsymbol{\lambda}^{(2)}_{k,n} = -1/2\mathbf{s}_{k,n}$ for the second case ($\mathbf{C} = \mathbf{F}^{-1}$). Thus, we can reformulate our problem into updating particles within the constrained domain, as addressed by Nguyen and Sakurai (2023). We define a strongly convex function φ as follows:

$$\varphi(\boldsymbol{\lambda}) = \frac{1}{2}\|\boldsymbol{\lambda}^{(1)}\|_2^2 + \langle \boldsymbol{\lambda}^{(2)}, \log \boldsymbol{\lambda}^{(2)} - 1\rangle,$$

where the log is taken elementwise. This function is composed of two terms: the first term keeps $\boldsymbol{\lambda}^{(1)}$ unchanged while the second term handles the non-negative constraint of $\boldsymbol{\lambda}^{(2)}$, which corresponds to the variance (see Beck and Teboulle (2003) or Appendix G for the background of the mirror descent). Then the mirror map induced by this convex function φ is defined as follows:

$$\nabla\varphi(\boldsymbol{\lambda}) = \boldsymbol{\zeta}, \tag{24}$$

where $\boldsymbol{\zeta} \in \mathbb{R}^d$ is defined as: $\boldsymbol{\zeta}^{(1)} = \boldsymbol{\lambda}^{(1)}$ and $\boldsymbol{\zeta}^{(2)} = \log \boldsymbol{\lambda}^{(2)}$. The inverse of the mirror map is defined as follows:

$$\nabla\varphi^*(\boldsymbol{\zeta}) = \boldsymbol{\lambda}, \tag{25}$$

where $\boldsymbol{\lambda}^{(1)} = \boldsymbol{\zeta}^{(1)}$, $\boldsymbol{\lambda}^{(2)} = \exp(\boldsymbol{\zeta}^{(2)})$ (with exp taken elementwise). The dual function of φ is denoted as φ^*. The basic idea of our solution is to map the parameters $\boldsymbol{\lambda}_{k,n}$ (for $k = 1, 2, ..., K$) to the dual space using the mirror map defined by (24) before each update. After updating these parameters in the given direction, we map them back to the original space using the inverse map defined by (25). This ensures that the updated parameters always belong to the constrained domain. In summary, we modify the updates as follows: for GFlowVI, we have

$$\begin{aligned}
&\mathbf{s}'_{k,n} = \log(\mathbf{s}_{k,n}).\\
&\mathbf{s}'_{k,n+1} = \mathbf{s}'_{k,n} - \eta \mathbf{w}_k \nabla_{\mathbf{s}_k} \log q(\mathbf{z}|\boldsymbol{\lambda}_{k,n}).\\
&+ \frac{\eta}{2} \oslash (\mathbf{s}_{k,n} \odot \mathbf{s}_{k,n}) \odot \mathtt{diag}\left[\nabla^2_{\mathbf{z}} f(\mathbf{z}) + \nabla^2_{\mathbf{z}} \log q_n(\mathbf{z})\right]\\
&\mathbf{s}_{k,n+1} = \exp(\mathbf{s}'_{k,n+1}).\\
&\boldsymbol{\mu}_{k,n+1} = \boldsymbol{\mu}_{k,n} - \eta\left[\nabla_{\mathbf{z}} f(\mathbf{z}) + \nabla_{\mathbf{z}} \log q_n(\mathbf{z})\right]\\
&- \eta \mathbf{w}_k \nabla_{\boldsymbol{\mu}_k} \log q(\mathbf{z}|\boldsymbol{\lambda}_{k,n}),
\end{aligned} \tag{26}$$

where the first line is to map the vectors $\mathbf{s}_{k,n}$ to the dual space through the mirror map (24), the second line is to update these vectors in the dual space, and the third line is to map the updated variance vectors back to the constrained domain through the inverse map (25). It can be confirmed that the variance vectors are always positive.

For NGFlowVI, we apply the same procedure to update the variance vectors. The modification of the update (23) can be expressed as follows:

$$\begin{aligned}
&\mathbf{s}'_{k,n} = \log(\mathbf{s}_{k,n}).\\
&\mathbf{s}'_{k,n+1} = \mathbf{s}'_{k,n} + \eta \mathtt{diag}\left[\nabla^2_{\mathbf{z}} f(\mathbf{z}) + \nabla^2_{\mathbf{z}} \log q_n(\mathbf{z})\right]\\
&- 2\eta \mathbf{w}_k (\mathbf{s}_{k,n} \odot \mathbf{s}_{k,n}) \odot \nabla_{\mathbf{S}_k} \log q(\mathbf{z}|\lambda_{k,n}).\\
&\mathbf{s}_{k,n+1} = \exp(\mathbf{s}'_{k,n+1}).\\
&\boldsymbol{\mu}_{k,n+1} = \boldsymbol{\mu}_{k,n} - \eta\left[\nabla_{\mathbf{z}} f(\mathbf{z}) + \nabla_{\mathbf{z}} \log q_n(\mathbf{z})\right] \oslash \mathbf{s}_{k,n+1}\\
&- \eta \mathbf{w}_k \nabla_{\boldsymbol{\mu}_k} \log q(\mathbf{z}|\boldsymbol{\lambda}_{k,n}) \oslash \mathbf{s}_{k,n+1}.
\end{aligned} \tag{27}$$

Remark 4. The approach outlined above can be generalized to handle the constraints imposed on $\boldsymbol{\lambda}^{(2)}$ such as $a \le \boldsymbol{\lambda}^{(2)} \le b$. We can modify the convex function φ as follows:

$$\begin{aligned}
\varphi(\boldsymbol{\lambda}) = \frac{1}{2}\|\boldsymbol{\lambda}^{(1)}\|_2^2 &+ \langle \boldsymbol{\lambda}^{(2)} - a, \log(\boldsymbol{\lambda}^{(2)} - a)\rangle\\
&+ \langle b - \boldsymbol{\lambda}^{(2)}, \log(b - \boldsymbol{\lambda}^{(2)})\rangle.
\end{aligned}$$

The mirror map in (24) for $\boldsymbol{\lambda}^{(2)}$ is modified as follows: $\boldsymbol{\zeta}^{(2)} = \log(\boldsymbol{\lambda}^{(2)} - a) - \log(b - \boldsymbol{\lambda}^{(2)})$. Also the inverse of the mirror map (25) for $\boldsymbol{\lambda}^{(2)}$ is modified as: $\boldsymbol{\lambda}^{(2)} = (b\exp(\boldsymbol{\zeta}^{(2)}) + a)/(\exp(\boldsymbol{\zeta}^{(2)}) + 1)$, thus, $\boldsymbol{\lambda}^{(2)}$ always satisfies the constraints during the optimization process, i.e., $a \le \boldsymbol{\lambda}^{(2)} \le b$.

4 Numerical Experiments

To validate and enhance our theoretical analyses of the proposed updates, we conduct a series of numerical experiments, including simulated experiments and applications to Bayesian neural networks. We compare GFlowVI and NGFlowVI with Wasserstein variational inference (WVI) (Lambert et al., 2022) and natural gradient variational inference for mixture models (NGVI) (Lin et al., 2019). We omit BBVI from our experiments, as previous work demonstrated that NGVI is superior to BBVI. We use numbers following the methods' names to denote the number of components K in the mixtures for each method. Experiments are done on a PC with Intel Core i9 and 64 GB memory.

Results on simulated datasets. We first consider to sample from a 4-cluster Gaussian mixture distribution π, defined on a two-dimensional space, with equal cluster weights. The objective is to minimize the KL

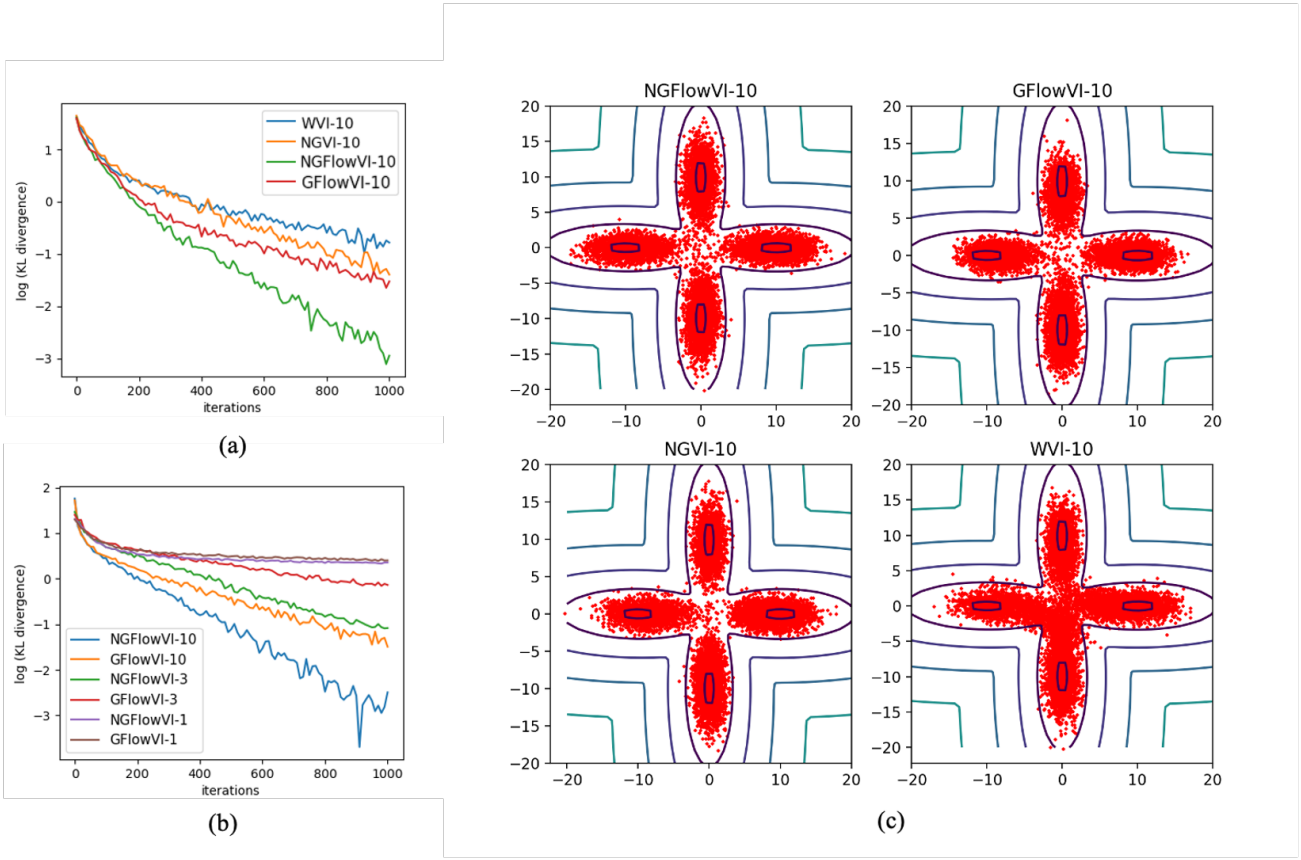

Figure 1: Experimental results on the synthetic dataset: (a) the estimated KL divergence in log scale between the target π and approximate density q over 1,000 iterations of four updates with $K = 10$; (b) performance of NGFlowVI and GFlowVI with varying values of K: 1, 3 and 5; (c) visualizations of 1,000 samples from q given by the four updates.

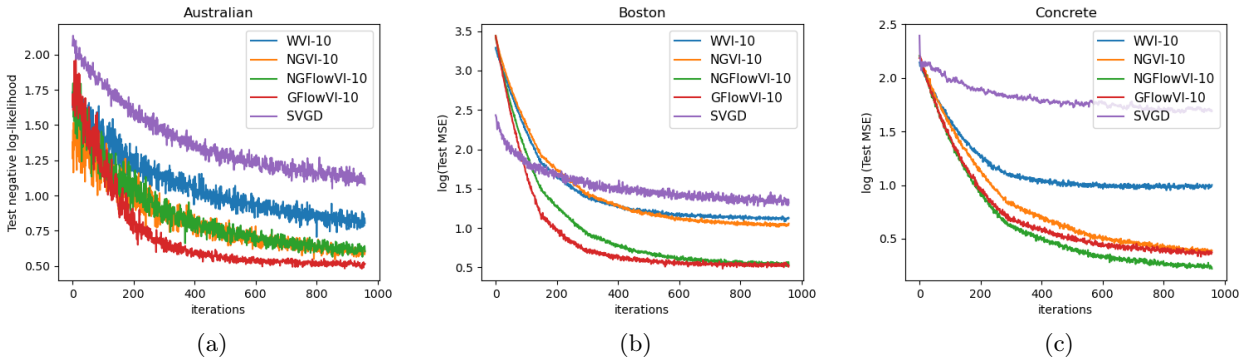

Figure 2: Average test negative log-likelihood of Bayesian neural networks (BNNs) on (a) 'Australia scale' and averaged test mean square error of BNNs on (b) 'Boston' and (c) 'Concrete' over 1000 iterations. For SVGD, 100 particles are used, while other methods approximate BNN weight posteriors with a Gaussian mixture ($K = 10$). Parameters are updated using WVI-10, NGVI-10, GFlowVI-10 and NGFlowVI-10. Results are averaged over 20 runs of 20 data splits.

divergence between q and π. For NGVI, GFlowVI and NGFlowVI, q is a mixture of diagonal Gaussians, while for WVI, q is a mixture of Gaussians with full covariance. Initially, the means of Gaussians are randomly sampled from a two-dimensional normal distribution and variances are set to the identity matrix. Means and covariance matrices are updated over 1,000 iterations with a fixed learning rate $\eta = 0.001$. The expectation $\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})}[\cdot]$ is estimated using a single sample.

Figure 1(a) shows the KL divergence between q and π over 1,000 iterations for the four updates when $K = 10$. The KL divergence is estimated by 10^4 MC samples from q. We see that NGFlowVI-10 converges faster than the others. GFlowVI-10's convergence is comparable to NGVI-10, and much faster than WVI-10. We also evaluate NGFlowVI and GFlowVI for $K = 1, 3, 10$, as shown in Figure 1(b). With $K = 1$, both NGFlowVI and GFlowVI perform poorly, as a single Gaussian might not capture the multi-modal target distribution. However, with $K = 10$, both methods effectively approximate the target. Figure 1(c) shows 1,000 samples from the approximate density q for WVI-10, NGVI-10, GFlowVI-10 and NGFlowVI-10. These evidences confirm the effectiveness of NGFlowVI and GFlowVI on this synthetic dataset. In addition, we apply these methods to approximate two other synthetic distributions defined on a two-dimensional space: a banana-shaped distribution and an X-shaped mixture of Gaussians. The densities of these distributions, the approximate KL divergence between q and the targets after 1000 particle updates, and the visualizations of 1000 samples are given in Table 2, Table 3 and Figure 3, respectively, in Appendix H.

Results on real-world datasets. We also validate our methods on Bayesian neural networks using real-world datasets and include SVGD (Liu and Wang, 2016)) for comparison. We use the following three datasets: 1) 'Australian': $N = 790$ examples , dimensionality= 14, with 345 for training; 2) 'Boston': $N = 506$, dimensionality= 8, with 455 for training; 3) 'Concrete': $N = 1030$, dimensionality= 13, with 927 for training. We perform classification on the first dataset and regression on the others, using 20 data splits provided by Gal and Ghahramani (2016). Results are averaged over 20 runs of these splits. We employ the same deep neural network architecture for all datasets with one hidden layer, 50 hidden units and ReLU activation. The regularization parameter and learning rate are set to 0.1 and 0.0001, respectively. We use minibatches of size 32 to approximate gradients and Hessians of f in (1). The posterior over the network weights is approximated by a Gaussian mixture with $K = 10$, and parameters are updated through 1,000 iterations. For predictions, we draw 100 samples of weights for the networks and calculate the average prediction for the given input. We use 100 particles for SVGD. We use 10 samples to estimate the expectation $\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})}[\cdot]$. Figure 2 shows the averaged negative log-likelihood over 1,000 iterations. GFlowVI-10 achieves the best convergence on the first two datasets, while NGFlowVI-10 and NGVI-10 leads on the third. See more details on the experiments in Appendix I.

5 Conclusions

We introduced a novel WGF-based approach for VI that operates on variational parameter domains, unlike previous methods, which focus on latent variable domains. This approach makes significant contributions to related fields. Our developed algorithms were empirically validated on the synthetic and real-world datasets, demonstrating their effectiveness. However, our current work is limited to diagonal Gaussian distributions for the algorithmic development. There are two main reasons. First, a single Gaussian may not adequately capture the complexity of the posterior (e.g. rotated Gaussian), but a mixture of diagonal Gaussian can provide a better approximation. Second, the proposed updates (16) might violate the parameter constraints, such as ensuring that the covariance matrix must be positive definite. To address this issue, we opted for the simpler case of diagonal covariance and employed the mirror descent for constrained optimization. We plan to address full covariance Gaussians in future work.

Acknowledgements

This work was supported in part by MEXT KAKENHI [grant number: 23K16939] (to D.H.N.) and MEXT KAKENHI [grant numbers: 22H03645 and 23K24901] (to H.M.).

References

Shun-Ichi Amari and Scott C Douglas. Why natural gradient? In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*, volume 2, pages 1213–1216. IEEE, 1998.

Amir Beck and Marc Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3): 167–175, 2003.

Rajendra Bhatia, Tanvi Jain, and Yongdo Lim. On the bures–wasserstein distance between positive definite matrices. *Expositiones Mathematicae*, 37(2):165–191, 2019.

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.

Georges Bonnet. Transformations des signaux aléatoires a travers les systemes non linéaires sans mémoire. In *Annales des Télécommunications*, volume 19, pages 203–220. Springer, 1964.

Kenneth F Caluya and Abhishek Halder. Gradient flow algorithms for density propagation in stochastic systems. *IEEE Transactions on Automatic Control*, 65(10):3991–4004, 2019.

Yongxin Chen, Tryphon T Georgiou, and Allen Tannenbaum. Optimal transport for gaussian mixture models. *IEEE Access*, 7:6269–6278, 2018.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

Thomas O Gallouët and Leonard Monsaingeon. A jko splitting scheme for kantorovich–fisher–rao gradient flows. *SIAM Journal on Mathematical Analysis*, 49 (2):1100–1130, 2017.

David Gunawan, Robert Kohn, and David Nott. Flexible variational bayes based on a copula of a mixture. *Journal of Computational and Graphical Statistics*, 33(2):665–680, 2024.

László Horváth. A refinement of the integral form of jensen's inequality. *Journal of Inequalities and Applications*, 2012:1–19, 2012.

Ya-Ping Hsieh, Chen Liu, and Volkan Cevher. Finding mixed nash equilibria of generative adversarial networks. In *International Conference on Machine Learning*, pages 2810–2819. PMLR, 2019.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37:183–233, 1999.

Richard Jordan, David Kinderlehrer, and Felix Otto. The variational formulation of the fokker–planck equation. *SIAM journal on mathematical analysis*, 29(1):1–17, 1998.

Mohammad Emtiyaz Khan and Didrik Nielsen. Fast yet simple natural-gradient descent for variational inference in complex models. In *2018 International Symposium on Information Theory and Its Applications (ISITA)*, pages 31–35. IEEE, 2018.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Marc Lambert, Sinho Chewi, Francis Bach, Silvère Bonnabel, and Philippe Rigollet. Variational inference via wasserstein gradient flows. *Advances in Neural Information Processing Systems*, 35:14434–14447, 2022.

Jen Ning Lim and Adam Johansen. Particle semi-implicit variational inference. *Advances in Neural Information Processing Systems*, 37:123954–123990, 2025.

Wu Lin, Mohammad Emtiyaz Khan, and Mark Schmidt. Fast and simple natural-gradient variational inference with mixture of exponential-family approximations. In *International Conference on Machine Learning*, pages 3992–4002. PMLR, 2019.

Dennis Victor Lindley. *Bayesian statistics: A review*. SIAM, 1972.

Chang Liu, Jingwei Zhuo, Pengyu Cheng, Ruiyi Zhang, and Jun Zhu. Understanding and accelerating particle-based variational inference. In *International Conference on Machine Learning*, pages 4082–4092. PMLR, 2019.

Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose bayesian inference algorithm. *Advances in neural information processing systems*, 29, 2016.

Dai Hai Nguyen and Tetsuya Sakurai. Mirror variational transport: a particle-based algorithm for distributional optimization on constrained domains. *Machine Learning*, 112(8):2845–2869, 2023.

Dai Hai Nguyen and Tetsuya Sakurai. Moreau-yoshida variational transport: a general framework for solving regularized distributional optimization problems. *Machine Learning*, 113(9):6697–6724, 2024.

Dai Hai Nguyen and Koji Tsuda. On a linear fused gromov-wasserstein distance for graph structured data. *Pattern Recognition*, 138:109351, 2023.

Dai Hai Nguyen, Canh Hao Nguyen, and Hiroshi Mamitsuka. Learning subtree pattern importance for weisfeiler-lehman based graph kernels. *Machine Learning*, 110:1585–1607, 2021.

Hermina Petric Maretic, Mireille El Gheche, Giovanni Chierchia, and Pascal Frossard. Got: an optimal transport framework for graph comparison. *Advances in Neural Information Processing Systems*, 32, 2019.

Robert Price. A useful theorem for nonlinear devices having gaussian inputs. *IRE Transactions on Information Theory*, 4(2):69–72, 1958.

Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Artificial intelligence and statistics*, pages 814–822. PMLR, 2014.

Ola Rønning, Eric Nalisnick, Christophe Ley, Padhraic Smyth, and Thomas Hamelryck. Elboing stein: Variational bayes with stein mixture inference. *arXiv preprint arXiv:2410.22948*, 2024.

Filippo Santambrogio. Optimal transport for applied mathematicians. *Birkäuser, NY*, 55(58-63):94, 2015.

Udo Von Toussaint. Bayesian inference in physics. *Reviews of Modern Physics*, 83(3):943, 2011.

Mingxuan Yi and Song Liu. Bridging the gap between variational inference and wasserstein gradient flows. *arXiv preprint arXiv:2310.20090*, 2023.

Mingzhang Yin and Mingyuan Zhou. Semi-implicit variational inference. In *International conference on machine learning*, pages 5660–5669. PMLR, 2018.

Longlin Yu and Cheng Zhang. Semi-implicit variational inference via score matching. *arXiv preprint arXiv:2308.10014*, 2023.

Chao Zhang, Zhijian Li, Hui Qian, and Xin Du. Dpvi: A dynamic-weight particle-based variational inference framework. *arXiv preprint arXiv:2112.00945*, 2021.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Not Applicable]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [[Not Applicable]]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Appendix for Wasserstein Gradient Flow over Variational Parameter Space for Variational Inference

A Proof of Theorem 1

Proof. To compute the first variation of $\mathcal{L}(p)$, suppose that $\varepsilon > 0$ and an arbitrary distribution $\chi \in \mathcal{P}(\Omega)$. We compute $(\mathcal{L}(\rho + \varepsilon\chi) - \mathcal{L}(\rho))/\varepsilon$ as follows:

$$
\begin{aligned}
&\frac{1}{\varepsilon}\left[\mathcal{L}(\rho+\varepsilon\chi) - \mathcal{L}(\rho)\right] = \\
&\frac{1}{\varepsilon}\int (\rho(\boldsymbol{\lambda}) + \varepsilon\chi(\boldsymbol{\lambda})) \int q(\mathbf{z}|\boldsymbol{\lambda}) \left[f(\mathbf{z}) + \log\left(\int (\rho(\boldsymbol{\lambda}) + \varepsilon\chi(\boldsymbol{\lambda}))\, q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}\right)\right] d\mathbf{z} d\boldsymbol{\lambda} \\
&-\frac{1}{\varepsilon}\int \rho(\boldsymbol{\lambda}) \int q(\mathbf{z}|\boldsymbol{\lambda}) \left[f(\mathbf{z}) + \log\left(\int \rho(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}\right)\right] d\mathbf{z} d\boldsymbol{\lambda} \\
&= \int \chi(\boldsymbol{\lambda}) \int q(\mathbf{z}|\boldsymbol{\lambda}) f(\mathbf{z}) d\mathbf{z} d\boldsymbol{\lambda} + \frac{1}{\varepsilon}\int (\rho(\boldsymbol{\lambda}) + \varepsilon\chi(\boldsymbol{\lambda})) \int q(z|\boldsymbol{\lambda}) \log\left(\int (\rho(\boldsymbol{\lambda}) + \varepsilon\chi(\boldsymbol{\lambda}))\, q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}\right) d\mathbf{z} d\boldsymbol{\lambda} \\
&-\frac{1}{\varepsilon}\int \rho(\boldsymbol{\lambda}) \int q(\mathbf{z}|\boldsymbol{\lambda}) \log\left(\int \rho(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}\right) d\mathbf{z} d\boldsymbol{\lambda} \\
&= \int \chi(\boldsymbol{\lambda}) \int q(\mathbf{z}|\boldsymbol{\lambda}) f(\mathbf{z}) d\mathbf{z} d\boldsymbol{\lambda} \\
&+\frac{1}{\varepsilon}\int (\rho(\boldsymbol{\lambda}) + \varepsilon\chi(\boldsymbol{\lambda})) \int q(\mathbf{z}|\boldsymbol{\lambda}) \log\left(\int (\rho(\boldsymbol{\lambda}) + \varepsilon\chi(\boldsymbol{\lambda}))\, q(z|\boldsymbol{\lambda}) d\boldsymbol{\lambda}\right) d\mathbf{z} d\boldsymbol{\lambda} - \\
&\frac{1}{\varepsilon}\int \rho(\boldsymbol{\lambda}) \int q(\mathbf{z}|\boldsymbol{\lambda}) \log\left(\int (\rho(\boldsymbol{\lambda}) + \varepsilon\chi(\boldsymbol{\lambda}))\, q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}\right) d\mathbf{z} d\boldsymbol{\lambda} \\
&+\frac{1}{\varepsilon}\left[\int \rho(\boldsymbol{\lambda}) \int q(\mathbf{z}|\boldsymbol{\lambda}) \log\left(\int (\rho(\boldsymbol{\lambda}) + \varepsilon\chi(\boldsymbol{\lambda}))\, q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}\right) d\mathbf{z} d\boldsymbol{\lambda} - \int \rho(\boldsymbol{\lambda}) \int q(\mathbf{z}|\boldsymbol{\lambda}) \log\left(\int \rho(\boldsymbol{\lambda}) q(z|\boldsymbol{\lambda}) d\boldsymbol{\lambda}\right) d\mathbf{z} d\boldsymbol{\lambda}\right] \\
&= \int \chi(\boldsymbol{\lambda}) \int q(\mathbf{z}|\boldsymbol{\lambda}) f(\mathbf{z}) d\mathbf{z} d\boldsymbol{\lambda} \\
&+ \underbrace{\int \chi(\boldsymbol{\lambda}) \int q(\mathbf{z}|\boldsymbol{\lambda}) \log\left(\int (\rho(\boldsymbol{\lambda}) + \varepsilon\chi(\boldsymbol{\lambda}))\, q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}\right) d\mathbf{z} d\boldsymbol{\lambda}}_{\text{(a)}} \\
&+ \underbrace{\frac{1}{\varepsilon}\left[\int \rho(\boldsymbol{\lambda}) \int q(\mathbf{z}|\boldsymbol{\lambda}) \log\left(1 + \frac{\varepsilon\int \chi(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}}{\int \rho(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}}\right) d\mathbf{z} d\boldsymbol{\lambda}\right]}_{\text{(b)}}.
\end{aligned}
$$

We process parts (a) and (b), when $\varepsilon \to 0$, as follows:

$$
\begin{aligned}
\lim_{\varepsilon\to 0} (\text{a}) &= \int \chi(\boldsymbol{\lambda}) \int q(\mathbf{z}|\boldsymbol{\lambda}) \log\left(\int \rho(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}\right) d\mathbf{z} d\boldsymbol{\lambda}, \\
\lim_{\varepsilon\to 0} (\text{b}) &= \int \rho(\boldsymbol{\lambda}) \int q(\mathbf{z}|\boldsymbol{\lambda}) \frac{\int \chi(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}}{\int \rho(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}} d\mathbf{z} d\boldsymbol{\lambda} = \int \int \rho(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda} \frac{\int \chi(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}}{\int \rho(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}} d\mathbf{z} \\
&= \int \chi(\boldsymbol{\lambda}) \int q(\mathbf{z}|\boldsymbol{\lambda}) d\mathbf{z} d\boldsymbol{\lambda} = \int \chi(\boldsymbol{\lambda}) d\boldsymbol{\lambda},
\end{aligned}
$$

where we have used the following equality for (b): $\lim_{\varepsilon\to 0}\frac{\log(1+\varepsilon x)}{\varepsilon}=x$ for all $x\in\mathbb{R}$.

So, we have:

$$\lim_{\varepsilon\to 0}\frac{1}{\varepsilon}\left[\mathcal{L}(\rho+\varepsilon\chi)-\mathcal{L}(\rho)\right]=\int\chi(\boldsymbol{\lambda})\left(\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})}\left[f(\mathbf{z})+\log q(\mathbf{z})\right]+1\right)d\boldsymbol{\lambda}.$$

By definition of the first variation of $\mathcal{L}$, this completes the proof of Theorem 1. □

B Proof of Proposition 2

Proof. Using the differential calculus in the Wasserstein space and the chain rule, we have:

$$\begin{aligned}\frac{d\mathcal{L}(\rho_t)}{dt}&=-\int\delta\mathcal{L}(\rho_t)(\boldsymbol{\lambda})\mathtt{div}(\rho_t\mathbf{C}v_t)d\boldsymbol{\lambda}=\int\langle\mathbf{C}(\boldsymbol{\lambda})v_t(\boldsymbol{\lambda}),\nabla_{\boldsymbol{\lambda}}\delta\mathcal{L}(\rho_t)(\boldsymbol{\lambda})\rangle d\rho_t(\boldsymbol{\lambda})\\&=-\int\langle v_t(\boldsymbol{\lambda}),\mathbf{C}(\boldsymbol{\lambda})v_t(\boldsymbol{\lambda})\rangle d\rho_t(\boldsymbol{\lambda}).\end{aligned}$$

□

C Proof of Proposition 3

Proof. Denote $v_n(\boldsymbol{\lambda})=-\nabla_{\boldsymbol{\lambda}}\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})}\left[f(\mathbf{z})+\log q_n(\mathbf{z})\right]$ where $q_n(\mathbf{z})=\mathbb{E}_{\boldsymbol{\lambda}\sim\rho_n}\left[q(\mathbf{z}|\boldsymbol{\lambda})\right]$, $\Phi_t(\boldsymbol{\lambda})=\boldsymbol{\lambda}+t\mathbf{C}(\boldsymbol{\lambda})v_n(\boldsymbol{\lambda})$ for $t\in[0,\eta]$, and $\nu_t=(\Phi_t)_{\#}\rho_n$. Then it is evident that $\nu_0=\rho_n$ and $\nu_\eta=\rho_{n+1}$.

We define $\phi(t)=\mathcal{L}(\nu_t)$. Clearly, $\phi(0)=\mathcal{L}(\rho_n)$ and $\phi(\eta)=\mathcal{L}(\rho_{n+1})$. Using a Taylor expansion, we have:

$$\phi(\eta)=\phi(0)+\eta\phi'(0)+\int_0^\eta(\eta-t)\phi''(t)dt. \tag{28}$$

Using the chain rule, we can estimate $\phi'(t)$ as follows:

$$\begin{aligned}\phi'(t)=&\frac{d}{dt}\int\rho_t(\boldsymbol{\lambda})\int q(\mathbf{z}|\boldsymbol{\lambda})\left[f(\mathbf{z})+\log\left(\int_{\boldsymbol{\lambda}}\rho_t(\boldsymbol{\lambda})q(\mathbf{z}|\boldsymbol{\lambda})d\boldsymbol{\lambda}\right)\right]d\mathbf{z}d\boldsymbol{\lambda}\\=&\frac{d}{dt}\int\rho_n(\boldsymbol{\lambda})\int q(\mathbf{z}|\Phi_t(\boldsymbol{\lambda}))\left[f(\mathbf{z})+\log\left(\int\rho_n(\boldsymbol{\lambda})q(\mathbf{z}|\Phi_t(\boldsymbol{\lambda}))d\boldsymbol{\lambda}\right)\right]d\mathbf{z}d\boldsymbol{\lambda}\\=&\int\rho_n(\boldsymbol{\lambda})\langle\frac{d\Phi_t(\boldsymbol{\lambda})}{dt},\nabla_{\boldsymbol{\lambda}}\int q(\mathbf{z}|\Phi_t(\boldsymbol{\lambda}))\left[f(\mathbf{z})+\log\left(\int\rho_n(\boldsymbol{\lambda})q(\mathbf{z}|\Phi_t(\boldsymbol{\lambda}))d\boldsymbol{\lambda}\right)\right]d\mathbf{z}\rangle d\boldsymbol{\lambda}\\=&\int\rho_n(\boldsymbol{\lambda})\langle\mathbf{C}(\boldsymbol{\lambda})v_n(\boldsymbol{\lambda}),\nabla_{\boldsymbol{\lambda}}\mathbb{E}_{\mathbf{z}\sim q(\cdot|\Phi_t(\boldsymbol{\lambda}))}\left[f(\mathbf{z})+\log q_t(\mathbf{z})\right]\rangle d\boldsymbol{\lambda},\end{aligned}$$

where $q_t(\mathbf{z})=\int\rho_n(\boldsymbol{\lambda})q(\mathbf{z}|\Phi_t(\boldsymbol{\lambda}))d\boldsymbol{\lambda}$. The second equality is obtained by applying the change of variable formula, and the last equality is obtained by the definition of v_n and Φ_t.

So, at $t=0$, we have:

$$\phi'(0)=-\int\langle v_n(\boldsymbol{\lambda}),\mathbf{C}(\boldsymbol{\lambda})v_n(\boldsymbol{\lambda})\rangle\rho_n(\boldsymbol{\lambda})d\boldsymbol{\lambda}=-\langle v_n,\mathbf{C}v_n\rangle_{L^2(p_n)}. \tag{29}$$

Next we estimate $\phi''(\boldsymbol{\lambda})$ as follows:

$$\begin{aligned}\phi''(\boldsymbol{\lambda})=&\frac{d}{dt}\phi'(t)=\int\rho_n(\boldsymbol{\lambda})\langle\mathbf{C}(\boldsymbol{\lambda})v_n(\boldsymbol{\lambda}),\frac{d}{dt}\nabla_{\boldsymbol{\lambda}}\mathbb{E}_{\mathbf{z}\sim q(\cdot|\Phi_t(\boldsymbol{\lambda}))}\left[f(\mathbf{z})+\log q_t(\mathbf{z})\right]\rangle d\boldsymbol{\lambda}\\=&\int\rho_n(\boldsymbol{\lambda})\langle\mathbf{C}(\boldsymbol{\lambda})v_n(\boldsymbol{\lambda}),\mathbf{H}_t(\boldsymbol{\lambda})v_n(\boldsymbol{\lambda})\rangle d\boldsymbol{\lambda}=\langle\mathbf{C}v_n,\mathbf{H}_tv_n\rangle_{L^2(p_n)},\end{aligned}$$

where $\mathbf{H}_t(\boldsymbol{\lambda}) = \nabla^2_{\boldsymbol{\lambda}} \mathbb{E}_{\mathbf{z}\sim q(\cdot|\Phi_t(\boldsymbol{\lambda}))} [f(\mathbf{z}) + \log q_t(\mathbf{z})]$. Now we need to upper-bound the operator norm of $\mathbf{H}_t$. Denote $\boldsymbol{\lambda}_t = \Phi_t(\boldsymbol{\lambda})$, we can rewrite $\mathbf{H}_t$ as follows:

$$\begin{aligned}\mathbf{H}_t(\boldsymbol{\lambda}) =& \nabla_{\boldsymbol{\lambda}} \left[\int \nabla_{\boldsymbol{\lambda}} q(\mathbf{z}|\boldsymbol{\lambda}_t) \left[f(\mathbf{z}) + \log q_t(\mathbf{z}) \right] d\mathbf{z} + \int q(\mathbf{z}|\boldsymbol{\lambda}_t) \nabla_{\boldsymbol{\lambda}} \log q_t(\mathbf{z}) d\mathbf{z} \right] \\ =& \int \nabla^2_{\boldsymbol{\lambda}} q(\mathbf{z}|\boldsymbol{\lambda}_t) \left[f(\mathbf{z}) + \log q_t(\mathbf{z}) \right] d\mathbf{z},\end{aligned}$$

where the second equality is obtained using the fact: $\nabla_{\boldsymbol{\lambda}} \log q_t(\mathbf{z}) = 0$. Therefore:

$$\begin{aligned}&\mathbf{H}_t(\boldsymbol{\lambda}) = \int \nabla_{\boldsymbol{\lambda}} \left[q(\mathbf{z}|\boldsymbol{\lambda}_t) \nabla_{\boldsymbol{\lambda}} \log q(\mathbf{z}|\boldsymbol{\lambda}_t) \right] \left[f(\mathbf{z}) + \log q_t(\mathbf{z}) \right] d\mathbf{z} \\ =& \int \left[\nabla_{\boldsymbol{\lambda}} q(\mathbf{z}|\boldsymbol{\lambda}_t) \right] \left[\nabla_{\boldsymbol{\lambda}} \log q(\mathbf{z}|\boldsymbol{\lambda}_t) \right]^{\intercal} \left[f(\mathbf{z}) + \log q_t(\mathbf{z}) \right] d\mathbf{z} + \int q(\mathbf{z}|\boldsymbol{\lambda}_t) \nabla^2_{\lambda} \log q(\mathbf{z}|\boldsymbol{\lambda}_t) \left[f(\mathbf{z}) + \log q_t(\mathbf{z}) \right] d\mathbf{z} \\ =& \int q(\mathbf{z}|\boldsymbol{\lambda}_t) \left[\nabla_{\boldsymbol{\lambda}} \log q(\mathbf{z}|\boldsymbol{\lambda}_t) \right] \left[\nabla_{\boldsymbol{\lambda}} \log q(\mathbf{z}|\boldsymbol{\lambda}_t) \right]^{\intercal} \left[f(\mathbf{z}) + \log q_t(\mathbf{z}) \right] d\mathbf{z} \\ &+ \int q(\mathbf{z}|\boldsymbol{\lambda}_t) \nabla^2_{\lambda} \log q(\mathbf{z}|\boldsymbol{\lambda}_t) \left[f(\mathbf{z}) + \log q_t(\mathbf{z}) \right] d\mathbf{z}.\end{aligned}$$

Then, the operator norm of $\mathbf{H}_t(\boldsymbol{\lambda})$ can be upper-bounded as follows:

$$\begin{aligned}&\|\mathbf{H}_t(\boldsymbol{\lambda})\|_{\text{op}} \leq \mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda}_t)} \|\nabla_{\boldsymbol{\lambda}} \log q(\mathbf{z}|\boldsymbol{\lambda}_t)\|_2^2 |f(\mathbf{z}) + \log q_t(\mathbf{z})| + \mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda}_t)} \|\nabla^2_{\boldsymbol{\lambda}} \log q(\mathbf{z}|\boldsymbol{\lambda}_t)\|_{\text{op}} |f(\mathbf{z}) + \log q_t(\mathbf{z})| \\ \leq& \left[\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda}_t)} \|\nabla_{\boldsymbol{\lambda}} \log q(\mathbf{z}|\boldsymbol{\lambda}_t)\|_2^2 \right] \left[\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda}_t)} |f(\mathbf{z})| + \mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda}_t)} |\log q_t(\mathbf{z})| \right] \\ &+ \left[\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda}_t)} \|\nabla^2_{\boldsymbol{\lambda}} \log q(\mathbf{z}|\boldsymbol{\lambda}_t)\|_{\text{op}} \right] \left[\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda}_t)} |f(\mathbf{z})| + \mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda}_t)} |\log q_t(\mathbf{z})| \right] \\ \leq& (\alpha + \beta)(M_1 + M_2),\end{aligned} \tag{30}$$

where the first inequality is obtained using the equality $\|\mathbf{a}\mathbf{b}^{\intercal}\|_{\text{op}} = \|\mathbf{a}\|_2 \|\mathbf{b}\|_2$ for two vectors $\mathbf{a}$ and $\mathbf{b}$; the second inequality is obtained by using the inequality $\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})} |h_1(\mathbf{z})||h_2(\mathbf{z})| \leq \mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})} |h_1(\mathbf{z})| \mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda})} |h_2(\mathbf{z})|$ for two scalar functions h_1 and h_2; the last inequality is obtained by using assumptions **(A1)**, **(A2)** and **(A3)**.

Thus, plugging the results (29) and (30) into (28), we can derive the following inequality:

$$\begin{aligned}\mathcal{L}(\rho_{n+1}) \leq& \mathcal{L}(\rho_n) - \eta \langle v_n, \mathbf{C} v_n \rangle_{L^2(\rho_n)} + \int_0^{\eta} (\eta - t) \kappa \langle v_n, \mathbf{C} v_n \rangle_{L^2(\rho_n)} dt \\ =& \mathcal{L}(\rho_n) - \eta \langle v_n, \mathbf{C} v_n \rangle_{L^2(\rho_n)} + \kappa \frac{\eta^2}{2} \langle v_n, \mathbf{C} v_n \rangle_{L^2(\rho_n)},\end{aligned}$$

which concludes the proof of Proposition 3. □

D Proof of Proposition 5

In this section, we provide the proof of Proposition 5. First we introduce the following useful lemmas.

Lemma 6. *Given two functions $g : \Omega \to \mathbb{R}$ and $h : \Omega \to \mathbb{R}$. We have:*

$$\int g(\boldsymbol{\lambda}) \log \frac{g(\boldsymbol{\lambda})}{h(\boldsymbol{\lambda})} d\boldsymbol{\lambda} \geq \left(\int g(\boldsymbol{\lambda}) d\boldsymbol{\lambda} \right) \log \frac{\int g(\boldsymbol{\lambda}) d\boldsymbol{\lambda}}{\int h(\boldsymbol{\lambda}) d\boldsymbol{\lambda}}.$$

We can generalize Lemma 6 above as follows:

Lemma 7. *Given two functions $g : \Omega \to \mathbb{R}$, $h : \Omega \to \mathbb{R}$ and $f : \mathbb{R} \to \mathbb{R}$ is a convex function. We have:*

$$\int g(\boldsymbol{\lambda}) f\left(\frac{h(\boldsymbol{\lambda})}{g(\boldsymbol{\lambda})} \right) d\boldsymbol{\lambda} \geq \left(\int g(\boldsymbol{\lambda}) d\boldsymbol{\lambda} \right) f\left(\frac{\int h(\boldsymbol{\lambda}) d\boldsymbol{\lambda}}{\int g(\boldsymbol{\lambda}) d\boldsymbol{\lambda}} \right).$$

Note that Lemma 6 can be trivially obtained by setting $f(u) = -\log u$ in Lemma 7.

Proof. Let $w(\boldsymbol{\lambda}) = g(\boldsymbol{\lambda})/\int_\Omega g(\boldsymbol{\lambda})d\boldsymbol{\lambda}$ and $u(\boldsymbol{\lambda}) = h(\boldsymbol{\lambda})/g(\boldsymbol{\lambda})$. Then by applying the continuous Jensen inequality (Horváth, 2012) for the convex function f, we have:

$$\int w(\boldsymbol{\lambda}) f(u(\boldsymbol{\lambda}))d\boldsymbol{\lambda} \geq f\left(\int w(\boldsymbol{\lambda})u(\boldsymbol{\lambda})d\boldsymbol{\lambda}\right),$$

which is equivalent to:

$$\int g(\boldsymbol{\lambda}) f\left(\frac{h(\boldsymbol{\lambda})}{g(\boldsymbol{\lambda})}\right) d\boldsymbol{\lambda} \geq \left(\int g(\boldsymbol{\lambda})d\boldsymbol{\lambda}\right) f\left(\int \frac{\cancel{g(\boldsymbol{\lambda})}}{\int g(\boldsymbol{\lambda})d\boldsymbol{\lambda}} \frac{h(\boldsymbol{\lambda})}{\cancel{g(\boldsymbol{\lambda})}} d\boldsymbol{\lambda}\right),$$

which concludes the proof of Lemma 7. □

Lemma 8. *Let $\mathcal{L}$ be defined in (1). For two distributions ρ', $\rho \in \mathcal{P}(\Omega)$, we have:*

$$\mathcal{L}(\rho') - \mathcal{L}(\rho) - \int \delta\mathcal{L}(\rho)(\boldsymbol{\lambda}) \left(\rho'(\boldsymbol{\lambda}) - \rho(\boldsymbol{\lambda})\right) d\boldsymbol{\lambda} \leq KL(\rho', \rho). \tag{31}$$

Proof. We can write $\mathcal{L}(\rho') - \mathcal{L}(\rho)$ as follows:

$$\mathcal{L}(\rho') - \mathcal{L}(\rho) = \int\int \rho'(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) \left[f(\mathbf{z}) + \log q'(\mathbf{z})\right] d\mathbf{z}d\boldsymbol{\lambda} - \int\int \rho(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) \left[f(\mathbf{z}) + \log q(\mathbf{z})\right] d\mathbf{z}d\boldsymbol{\lambda},$$

where $q'(\mathbf{z}) = \int \rho'(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}$ and $q(\mathbf{z}) = \int \rho(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) d\boldsymbol{\lambda}$.

Thus, the left-hand side (LHS) of (31) can be rewritten as follows:

$$\text{LHS of (31)} = \int\int \rho'(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) \log \frac{q'(\mathbf{z})}{q(\mathbf{z})}) d\mathbf{z}d\boldsymbol{\lambda} = \int q'(\mathbf{z}) \log \frac{q'(\mathbf{z})}{q(\mathbf{z})}) d\mathbf{z}. \tag{32}$$

Applying Lemma 6 by setting $g(\boldsymbol{\lambda}) = \rho'(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda})$ and $h(\boldsymbol{\lambda}) = \rho(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda})$, we have:

$$q'(\mathbf{z}) \log \frac{q'(\mathbf{z})}{q(\mathbf{z})} \leq \int \rho'(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) \log \frac{\rho'(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda})}{\rho(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda})} d\boldsymbol{\lambda} = \int \rho'(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) \log \frac{\rho'(\boldsymbol{\lambda})}{\rho(\boldsymbol{\lambda})} d\boldsymbol{\lambda}. \tag{33}$$

Thus, using (32), we have:

$$\begin{aligned}\text{LHS of (31)} = \int\int \rho'(\boldsymbol{\lambda}) q(\mathbf{z}|\boldsymbol{\lambda}) \log \frac{\rho'(\boldsymbol{\lambda})}{\rho(\boldsymbol{\lambda})} d\boldsymbol{\lambda}d\mathbf{z} &= \int \rho'(\boldsymbol{\lambda}) \log \frac{\rho'(\boldsymbol{\lambda})}{\rho(\boldsymbol{\lambda})} \left(\int q(\mathbf{z}|\boldsymbol{\lambda}) d\mathbf{z}\right) d\boldsymbol{\lambda} \\ &= \int \rho'(\boldsymbol{\lambda}) \log \frac{\rho'(\boldsymbol{\lambda})}{\rho(\boldsymbol{\lambda})} d\boldsymbol{\lambda},\end{aligned}$$

which concludes the proof of Lemma 8. □

Now we are ready for giving the proof of Proposition 5.

Proof. For the first update of (16), we obtain the following inequality by Proposition 3:

$$\mathcal{L}(\bar{\rho}_n) - \mathcal{L}(\rho_n) \leq -\eta \left(1 - \kappa\frac{\eta}{2}\right) \langle v_n, \mathbf{C}v_n\rangle_{L^2(\rho_n)}. \tag{34}$$

We next consider the second update of (16). As $\rho_{n+1} = \mathrm{MD}_\eta(\bar{\rho}_n, \delta\mathcal{L}(\bar{\rho}_n))$, the first-order optimality condition yields:

$$\eta\delta\mathcal{L}(\bar{\rho}_n) = -\log \rho_{n+1} + \log \bar{\rho}_n + \text{constant} \tag{35}$$

Thus, we have:

$$\begin{aligned}&\int \delta\mathcal{L}(\bar{\rho}_n)(\boldsymbol{\lambda})(\rho(\boldsymbol{\lambda}) - \bar{\rho}_n(\boldsymbol{\lambda}))d\boldsymbol{\lambda} \\ &= \int \delta\mathcal{L}(\bar{\rho}_n)(\boldsymbol{\lambda})(\rho_{n+1}(\boldsymbol{\lambda}) - \bar{\rho}_n(\boldsymbol{\lambda}))d\boldsymbol{\lambda} + \int \delta\mathcal{L}(\bar{\rho}_n)(\boldsymbol{\lambda})(\rho(\boldsymbol{\lambda}) - \rho_{n+1}(\boldsymbol{\lambda}))d\boldsymbol{\lambda} \\ &= \int \delta\mathcal{L}(\bar{\rho}_n)(\boldsymbol{\lambda})(\rho_{n+1}(\boldsymbol{\lambda}) - \bar{\rho}_n(\boldsymbol{\lambda}))d\boldsymbol{\lambda} + \frac{1}{\eta}\int (\log \bar{\rho}_n(\boldsymbol{\lambda}) - \log \rho_{n+1}(\boldsymbol{\lambda}))(\rho(\boldsymbol{\lambda}) - \rho_{n+1}(\boldsymbol{\lambda}))d\boldsymbol{\lambda} \\ &= \int \delta\mathcal{L}(\bar{\rho}_n)(\boldsymbol{\lambda})(\rho_{n+1}(\boldsymbol{\lambda}) - \bar{\rho}_n(\boldsymbol{\lambda}))d\boldsymbol{\lambda} + \frac{1}{\eta}\left[KL(\rho_{n+1}, \bar{\rho}_n) + \mathrm{KL}(\rho, \rho_{n+1}) - \mathrm{KL}(\rho, \bar{\rho}_n)\right],\end{aligned}$$

where the second equality is obtained by applying (35) and the third equality is obtained by three-point identity for the KL divergence. Taking $\rho = \bar{\rho}_n$, we have:

$$\int \delta\mathcal{L}(\bar{\rho}_n)(\boldsymbol{\lambda})(\rho_{n+1}(\boldsymbol{\lambda}) - \bar{\rho}_n(\boldsymbol{\lambda}))d\boldsymbol{\lambda} = -\frac{1}{\eta}\mathrm{KL}(\rho_{n+1}, \bar{\rho}_n) - \frac{1}{\eta}\mathrm{KL}(\bar{\rho}_n, \rho_{n+1}). \tag{36}$$

Using (31) and (36), we have:

$$\begin{aligned}\mathcal{L}(\rho_{n+1}) - \mathcal{L}(\bar{\rho}_n) &\leq \int \delta\mathcal{L}(\bar{\rho}_n)(\boldsymbol{\lambda})\left(\rho_{n+1}(\boldsymbol{\lambda}) - \bar{\rho}_n(\boldsymbol{\lambda})\right) d\boldsymbol{\lambda} + \mathrm{KL}(\rho_{n+1}, \bar{\rho}_n) \\ &= -\left(\frac{1}{\eta} - 1\right) KL(\rho^{n+1}, \rho^n) - KL(\rho^n, \rho^{n+1}).\end{aligned}$$

Combining with (34), we conclude the proof of Proposition 5.

□

E The updates (16) viewed as the *preconditioned* Wasserstein-Fisher-Rao gradient flow of $\mathcal{L}$

Below, we demonstrate that the updates (16) are indeed related to the Wasserstein-Fisher-Rao gradient flow of $\mathcal{L}$ in the limit as $\eta \to 0$. First, it is known in (Gallouët and Monsaingeon, 2017, Eq 2.6) that the gradient flow of $\mathcal{L}(\rho_t)$ with respect to the Fisher-Rao distance is given by:

$$\frac{\partial \rho_t(\boldsymbol{\lambda})}{\partial t} = -\delta\mathcal{L}(\rho_t)(\boldsymbol{\lambda})\rho_t(\boldsymbol{\lambda}). \tag{37}$$

Second, we show that the second update of (16) to update the weights of particles can be viewed as the Fisher-Rao gradient flow of $\mathcal{L}$ as $\eta \to 0$. We consider the mirror descent update formula (see (4) and (16)). By taking the first variation and setting it to zero, we obtain:

$$\delta\mathcal{L}(\rho_n)(\boldsymbol{\lambda}) + \frac{1}{\eta}\log\frac{\rho_{n+1}(\boldsymbol{\lambda})}{\rho_n(\boldsymbol{\lambda})} = C, \tag{38}$$

for some constant $C > 0$.Therefore,

$$\rho_{n+1}(\boldsymbol{\lambda}) = \frac{\rho_n(\boldsymbol{\lambda})\exp\left(-\eta\delta\mathcal{L}(\rho_n)(\boldsymbol{\lambda})\right)}{\int \rho_n(\boldsymbol{\lambda})\exp\left(-\eta\delta\mathcal{L}(\rho_n)(\boldsymbol{\lambda})\right) d\boldsymbol{\lambda}}. \tag{39}$$

As $\eta \to 0$, we can approximate $\exp(-\eta x) \approx 1 - \eta x + O(\eta^2)$. Thus,

$$\rho_{n+1}(\boldsymbol{\lambda}) = \frac{\rho_n(\boldsymbol{\lambda})\left[1 - \eta\delta\mathcal{L}(\rho_n)(\boldsymbol{\lambda}) + O(\eta^2)\right]}{\int \rho_n(\boldsymbol{\lambda})\left[1 - \eta\delta\mathcal{L}(\rho_n)(\boldsymbol{\lambda}) + O(\eta^2)\right] d\boldsymbol{\lambda}} = \frac{\rho_n(\boldsymbol{\lambda})\left[1 - \eta\delta\mathcal{L}(\rho_n)(\boldsymbol{\lambda}) + O(\eta^2)\right]}{1 - \eta\left[1 + \mathcal{L}(\rho_n)\right] + O(\eta^2)}, \tag{40}$$

where we have used $\int \rho_n(\boldsymbol{\lambda})d\boldsymbol{\lambda} = 1$ and $\int \rho_n(\boldsymbol{\lambda})\delta\mathcal{L}(\rho_n)(\boldsymbol{\lambda})d\boldsymbol{\lambda} = \mathcal{L}(\rho_n) + 1$. Therefore,

$$\rho_{n+1}(\boldsymbol{\lambda}) = \rho_n(\boldsymbol{\lambda})\left[1 - \eta\delta\mathcal{L}(\rho_n)(\boldsymbol{\lambda}) + O(\eta^2)\right], \tag{41}$$

which leads to:

$$\frac{\rho_{n+1}(\boldsymbol{\lambda}) - \rho_n(\boldsymbol{\lambda})}{\eta} = \delta\mathcal{L}(\rho_n)(\boldsymbol{\lambda})\rho_n(\boldsymbol{\lambda}). \tag{42}$$

In other words, the continuous time limit of the mirror descent update is:

$$\frac{\partial \rho_t(\boldsymbol{\lambda})}{\partial t} = -\delta\mathcal{L}(\rho_t)(\boldsymbol{\lambda})\rho_t(\boldsymbol{\lambda}),$$

which is identical to the Fisher-Rao flow of $\mathcal{L}$.

In summary, the first update of (16) corresponds to the preconditioned Wasserstein gradient flow, while the second one aligns with the Fisher-Rao gradient flow as $\eta \to 0$. Thus, the proposed update (16) can be viewed as the discrete approximation of the preconditioned Wasserstein-Fisher-Rao flow of $\mathcal{L}$.

F Derivation of GFlowVI and NGFlowVI for Diagonal Gaussian Variance Inference

In this section we derive updates for GFlowVI and NGFlowVI. For the first case $\mathbf{C} = \mathbf{I}$, let recall $\boldsymbol{\lambda}_{k,n} = (\boldsymbol{\lambda}_{k,n}, \mathbf{s}_{k,n})$ and mean $\boldsymbol{\lambda}_{k,n}$ and vector $\mathbf{s}_{k,n}$ are updated as follows (see (16)):

$$\begin{aligned}\boldsymbol{\mu}_{k,n+1} =& \boldsymbol{\mu}_{k,n} - \eta\nabla_{\boldsymbol{\mu}_k}\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda}_{k,n})}\left[f(\mathbf{z}) + \log q_n(\mathbf{z})\right].\\ \mathbf{s}_{k,n+1} =& \mathbf{s}_{k,n} - \eta\nabla_{\mathbf{s}_k}\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda}_{k,n})}\left[f(\mathbf{z}) + \log q_n(\mathbf{z})\right].\end{aligned}$$

We denote $G = \mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda}_{k,n})}\left[f(\mathbf{z}) + \log q_n(\mathbf{z})\right]$. We can estimate the gradients of G with respect to $\boldsymbol{\mu}_k$ and $\mathbf{s}_k$ as follows:

$$\begin{aligned}\nabla_{\boldsymbol{\mu}_k} G =& \int \nabla_{\boldsymbol{\mu}_k} q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\left[f(\mathbf{z}) + \log q_n(\mathbf{z})\right]d\mathbf{z} + \int q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\nabla_{\boldsymbol{\mu}_k}\log q_n(\mathbf{z})d\mathbf{z}\\ =& -\int \nabla_{\mathbf{z}} q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\left[f(\mathbf{z}) + \log q_n(\mathbf{z})\right]d\mathbf{z} + \int q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\frac{q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\nabla_{\boldsymbol{\mu}_k}\log q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})}{q_n(\mathbf{z})}d\mathbf{z}\\ =& \int q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\left[\nabla_{\mathbf{z}} f(\mathbf{z}) + \nabla_{\mathbf{z}}\log q_n(\mathbf{z})\right]d\mathbf{z} + \int q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\mathbf{w}_k(\mathbf{z})\nabla_{\boldsymbol{\mu}_k}\log q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\\ =& \mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda}_{k,n})}\left[\nabla_{\mathbf{z}} f(\mathbf{z}) + \nabla_{\mathbf{z}}\log q_n(\mathbf{z}) + \mathbf{w}_k(\mathbf{z})\nabla_{\boldsymbol{\mu}_k}\log q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\right],\end{aligned} \tag{43}$$

where $\mathbf{w}_k(\mathbf{z}) = q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})/q_n(\mathbf{z})$. In the second equality, we have used the identity $\nabla_{\boldsymbol{\mu}} q(\mathbf{z}|\boldsymbol{\lambda}) = -\nabla_{\mathbf{z}} q(\mathbf{z}|\boldsymbol{\lambda})$ for q being a Gaussian distribution; in the third equality, we have used the integration by parts for the first term and $\nabla_{\boldsymbol{\mu}} q(\mathbf{z}|\boldsymbol{\lambda}) = q(\mathbf{z}|\boldsymbol{\lambda})\nabla_{\boldsymbol{\mu}}\log q(\mathbf{z}|\boldsymbol{\lambda})$ for the second term.

$$\begin{aligned}&\nabla_{\mathbf{s}_k} G = -1 \oslash (\mathbf{s}_{k,n}\odot\mathbf{s}_{k,n})\odot\int \nabla_{\boldsymbol{\sigma}_k^2} q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\left[f(\mathbf{z}) + \log q_n(\mathbf{z})\right]d\mathbf{z}\\ &+\int q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\nabla_{\mathbf{s}_k}\log q_n(\mathbf{z})d\mathbf{z}\\ &= -1\oslash(\mathbf{s}_{k,n}\odot\mathbf{s}_{k,n})\odot\int q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\frac{1}{2}\mathtt{diag}\left[\nabla_{\mathbf{z}}^2 f(\mathbf{z}) + \nabla_{\mathbf{z}}^2\log q_n(\mathbf{z})\right]d\mathbf{z}\\ &+\int q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\frac{q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\nabla_{\mathbf{s}_k}\log q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})}{q_n(\mathbf{z})}d\mathbf{z}\\ &=\mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda}_{k,n})}\left[-\frac{1}{2}\oslash(\mathbf{s}_{k,n}\odot\mathbf{s}_{k,n})\odot\mathtt{diag}\left[\nabla_{\mathbf{z}}^2 f(\mathbf{z}) + \nabla_{\mathbf{z}}^2\log q_n(\mathbf{z})\right] + \mathbf{w}_k(\mathbf{z})\nabla_{\mathbf{s}_k}\log q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\right],\end{aligned} \tag{44}$$

where we have used the change of variable formula in the first equation, the relation (21) for the first term of the second equation. By drawing a sample $\mathbf{z}$ from $q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})$, we have the update of GFlowVI (20).

Next we derive the update of NGFlowVI ($\mathbf{C} = \mathbf{F}^{-1}$). We consider $\boldsymbol{\lambda}_k$ to be the natural parameter of the diagonal Gaussian $q(\mathbf{z}|\boldsymbol{\lambda}_k)$. Specifically, the natural parameters and expectation parameters of the k-th Gaussian at the n-th iteration can be defined as follows:

$$\begin{aligned}&\boldsymbol{\lambda}_{k,n}^{(1)} = \mathbf{s}_{k,n}\odot\boldsymbol{\mu}_{k,n}, \boldsymbol{\lambda}_{k,n}^{(2)} = -\frac{1}{2}\mathbf{s}_{k,n},\\ &\mathbf{m}_{k,n}^{(1)} = \boldsymbol{\mu}_{k,n}, \mathbf{m}_{k,n}^{(2)} = \boldsymbol{\mu}_{k,n}\odot\boldsymbol{\mu}_{k,n} + 1\oslash\mathbf{s}_{k,n}.\end{aligned}$$

Table 1: Illustration on effect of the MD iterates on the average prediction losses of BNNs on three datasets: 'Australian scale' (negative log-likelihood), 'Boston' and 'Concrete' (mean square error). The results compare the the average prediction losses after 1000 iterations of GFlowVI and NGFlowVI (with MD iterates) against their counterparts without MD iterates (w/o-MD), demonstrating that incorporating MD iterates enhances prediction accuracy.

Methods	Australian	Boston	Concrete
GFlowVI	0.51±0.02	1.73±0.28	1.49±0.08
GFlowVI-w/o-MD	0.52±0.05	1.81±0.15	1.74±0.03
NGFlowVI	0.6±0.03	1.71±0.09	1.25±0.06
NGFlowVI-w/o-MD	0.72±0.05	1.87±0.15	1.34±0.11

Then, the natural parameters are updated as follows (by (3) in the main text):

$$\boldsymbol{\lambda}_{k,n+1} = \boldsymbol{\lambda}_{k,n} - \eta \nabla_{\mathbf{m}_k} G.$$

Using the chain rule (see Appendix B.1 in Khan and Nielsen (2018)), we can express the gradients of G with respect to expectation parameter $\mathbf{m}_k$ in terms of the gradients with respect to $\boldsymbol{\mu}_k$ and $\boldsymbol{\sigma}_k^2$ as follows:

$$\nabla_{\mathbf{m}_k^{(1)}} G = \nabla_{\boldsymbol{\mu}_k} G - 2 \left[\nabla_{\boldsymbol{\sigma}_k^2} G\right] \boldsymbol{\mu}_k, \nabla_{\mathbf{m}_k^{(2)}} G = \nabla_{\boldsymbol{\sigma}_k^2} G.$$

By following the derivation in Khan and Nielsen (2018), the natural-gradient update is simplified as follows:

$$\mathbf{s}_{k,n+1} = \mathbf{s}_{k,n} + 2\eta \left[\nabla_{\boldsymbol{\sigma}_k^2} G\right]. \tag{45}$$

$$\boldsymbol{\mu}_{k,n+1} = \boldsymbol{\mu}_{k,n} - \eta \left[\nabla_{\boldsymbol{\mu}_k} G\right] \oslash \mathbf{s}_{k,n+1}. \tag{46}$$

Using (44), we can derive the full update for $\mathbf{s}_{k,n}$ in (45) as follows:

$$\begin{aligned}\mathbf{s}_{k,n+1} =& \mathbf{s}_{k,n} - 2\eta(\mathbf{s}_{k,n} \odot \mathbf{s}_{k,n}) \left[\nabla_{\mathbf{s}_k} G\right] \\ =& \mathbf{s}_{k,n} + \mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda}_{k,n})} \left[\eta \mathtt{diag}\left[\nabla_{\mathbf{z}}^2 f(\mathbf{z}) + \nabla_{\mathbf{z}}^2 \log q_n(\mathbf{z})\right] - 2\eta \mathbf{w}_k(\mathbf{z})(\mathbf{s}_{k,n} \odot \mathbf{s}_{k,n}) \odot \nabla_{\mathbf{s}_k} \log q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\right]\end{aligned}$$

Lastly, using (43), we can derive the full update for $\boldsymbol{\mu}_{k,n}$ in (46) as follows:

$$\begin{aligned}\boldsymbol{\mu}_{k,n+1} =& \boldsymbol{\mu}_{k,n} - \eta \left[\nabla_{\boldsymbol{\mu}_k} G\right] \oslash \mathbf{s}_{k,n+1} \\ =& \boldsymbol{\mu}_{k,n} - \eta \mathbb{E}_{\mathbf{z}\sim q(\cdot|\boldsymbol{\lambda}_{k,n})} \left[\nabla_{\mathbf{z}} f(\mathbf{z}) + \nabla_{\mathbf{z}} \log q_n(\mathbf{z}) + \mathbf{w}_n(\mathbf{z}) \nabla_{\boldsymbol{\mu}_k} \log q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})\right] \oslash \mathbf{s}_{k,n+1}.\end{aligned}$$

By drawing a sample $\mathbf{z}$ from $q(\mathbf{z}|\boldsymbol{\lambda}_{k,n})$, we derive the update of NGFlowVI (23).

G Background on the Mirror Descent Algorithm

We provide a brief background on the mirror descent (**MD**) algorithm for optimization. Suppose we wish to minimize a function over a domain $\mathcal{X}$, say $\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x})$. When $\mathcal{X}$ is unconstrained, gradient descent is the standard algorithm to optimize f by solving the following optimization problem for each step t:

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}\in\mathcal{X}} \langle \nabla f(\mathbf{x}_t), \mathbf{x}\rangle + \frac{1}{2\eta_t} \|\mathbf{x} - \mathbf{x}_t\|_2^2. \tag{47}$$

To deal with the constrained optimization problems, the Mirror Descent (**MD**) algorithm replaces $\|\cdot\|_2$ in (47) with a function φ that reflects the geometry of the problem (Beck and Teboulle, 2003). The **MD** algorithm chooses Φ to be the Bregman divergence induced by a strongly convex function $\varphi : \mathcal{X} \to \mathbb{R}$ as follows: $\Phi(\mathbf{x}', \mathbf{x}) = \varphi(\mathbf{x}') - \varphi(\mathbf{x}) - \langle \nabla\varphi(\mathbf{x}), \mathbf{x}' - \mathbf{x}\rangle$ for $\mathbf{x}', \mathbf{x} \in \mathcal{X}$. Then, the solution of (47) for each step becomes:

$$\mathbf{x}_{t+1} = \nabla\varphi^* \left(\nabla\varphi(\mathbf{x}_t) - \eta_t \nabla f(\mathbf{x}_t)\right), \tag{48}$$

where $\varphi^*(\mathbf{y}) = \sup_{\mathbf{x}\in\mathcal{X}} \langle \mathbf{x}, \mathbf{y}\rangle - \varphi(\mathbf{x})$ is the convex conjugate of function φ and $\nabla\varphi^*(\mathbf{y}) = (\nabla\varphi)^{-1}(\mathbf{y})$ is the inverse map. Intuitively, the **MD** update (48) is composed of three steps: 1) mapping $\mathbf{x}_t$ to $\mathbf{y}_t$ by $\nabla\varphi$, 2) applying the update: $\mathbf{y}_{k+1} = \mathbf{y}_t - \eta_t \nabla f(\mathbf{x}_t)$, and 3) mapping back through $\mathbf{x}_{t+1} = \nabla\varphi^*(\mathbf{y}_{t+1})$.

Table 2: Banana-shaped distribution and X-shaped mixture of Gaussians.

Name	$\pi(\mathbf{z})$	**Parameters**
Banana-shaped	$\mathbf{z} = \left(v_1, v_1^2 + v_2 + 1\right), \mathbf{v} \sim \mathcal{N}(0, \Sigma)$	$\Sigma = [[1, 0.9], [0.9, 1]]\ /0.19$
X-shaped	$0.5\mathcal{N}(\mathbf{z}\|0, \Sigma_1) + 0.5\mathcal{N}(\mathbf{z}\|0, \Sigma_2)$	$\Sigma_1 = [[2, 1.8], [1.8, 2]]\ /0.76$,$\Sigma_2 = [[2, 1.8], [1.8, 2]]\ /0.76$

Table 3: The approximate KL divergence between the targets and q using 1000 updates of particles, averaged over five runs.

Targets	**WVI**	**NGVI**	**GFlowVI**	**NGFlowVI**
Banana-shaped	0.15±0.02	0.32±0.01	0.21±0.02	0.12±0.02
X-shaped	0.03±0.02	0.05±0.03	0.04±0.05	0.02±0.02

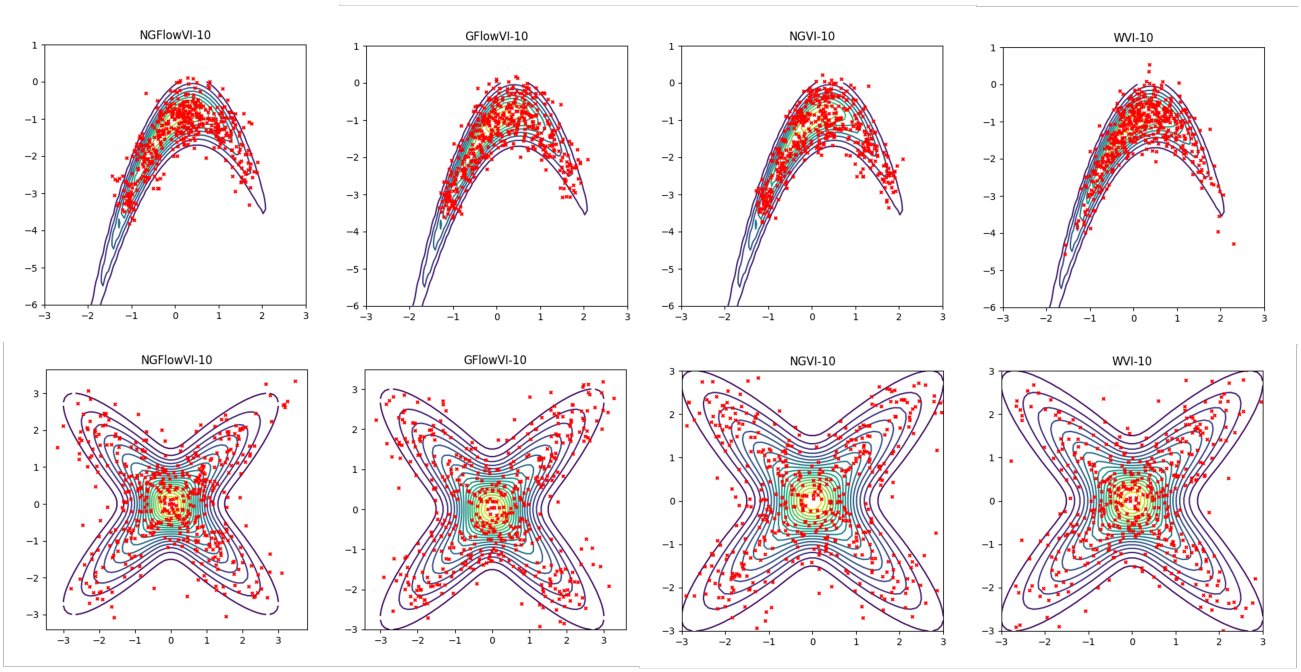

Figure 3: Experimental results on two synthetic datasets with visualization of 1000 samples from the variational distribution q produced by four methods: NGFlowVI, GFlowVI, NGVI and WVI, using $K = 10$ particles. These samples are used to approximate two target distributions: Banana-shaped distribution (the first row) and X-shaped distribution (the second row).

Table 4: Average prediction losses of BNNs on three datasets: 'Australian scale' (negative log-likelihood), 'Boston' and 'Concrete' (mean square error). The best values are indicated in bold.

Methods	Australian	Boston	Concrete
SVGD	1.13 ± 0.04	3.78 ± 0.22	6.44 ± 0.92
D-Blob-CA	1.14 ± 0.03	3.24 ± 0.37	4.71 ± 1.25
D-GFSD-CA	0.98 ± 0.04	3.47 ± 0.43	5.22 ± 1.02
WVI-10	0.85 ± 0.05	3.06 ± 0.17	2.71 ± 0.14
NGVI-10	0.62 ± 0.06	2.72 ± 0.16	1.49 ± 0.05
GFlowVI-10	$\mathbf{0.51} \pm 0.02$	1.73 ± 0.28	1.49 ± 0.08
NGFlowVI-10	0.61 ± 0.03	$\mathbf{1.71} \pm 0.09$	$\mathbf{1.25} \pm 0.06$

Table 5: Average running times (in seconds) required for one epoch of methods: SVGD, WVI, NGVI, GFlowVI and NGFlowVI on three datasets: 'Australian scale', 'Boston' and 'Concrete'.

Methods	Australian	Boston	Concrete
SVGD(m=50)	3.77±0.07	5.23±0.02	10.37±0.04
SVGD(m=100)	15.19±0.14	20.09±0.11	39.93±0.33
SVGD(m=500)	362.23±9.5	321.01±10.01	557.12±18.55
WVI	79.49±6.04	75.61±1.43	61.07±0.28
NGVI	6.55±0.24	9.46±0.19	19.69±0.32
GFlowVI	6.27±0.13	8.19±0.32	15.73±0.11
NGFlowVI	6.22±0.05	7.98±0.17	16.79±0.38

H Additional Experimental Results on Synthetic Datasets

In this section, we consider to sample from two other synthetic distributions defined on a two-dimensional space: a banana-shaped distribution and an X-shaped mixture of Gaussian. The densities of these distributions are given in Table 2.

We compare WVI, NGVI, GFlowVI, and NGFlowVI, using $K = 10$ particles. For these target distributions, the objective of compared methods is to produce a variational distribution q that closely approximates the target distribution. Table 3 presents the approximate KL divergence between q and the target distributions, using 1000 particle updates, averaged over five runs. Figure 3 illustrates 1000 samples from variational distributions fitted by compared methods align well with the shapes of target distributions.

I Additional Experimental Results on Real-world datasets

Effects of infinite-dimensional MD. We examined two scenarios for GFlow and NGFlow (both with K=10): one that updates both $\boldsymbol{\lambda}_k$ and a_k, for $k = 1, ..., K$ (denoted as GFlowVI and NGFlowVI, respectively), and another that updates $\boldsymbol{\lambda}_k$, for $k = 1, ..., K$, while keeping the weights fixed at 1/K (denoted by suffix w/o-MD). The results, shown in Table 1, demonstrate that the weight update scheme using MD iterates improves performance.

Average prediction loss. We compare our methods GFlowVI and NGFlowVI to WVI, NGVI and SVGD in terms of average prediction loss. Further, for weighting particles, we consider two DPVI algorithms (Zhang et al., 2021): D-Blob-CA and DP-GFSD-CA. Like SVGD, these methods approximate the gradient of log of empirical distribution using kernels, but they dynamically adjust particle weights. We use RBF kernel and the median method (Liu and Wang, 2016) for SVGD, D-Blob-CA and DP-GFSD-CA, and represent q with 100 samples (in latent space). We report the results after 1000 particle updates in Table 4. We observe that the DPVI methods

Table 6: Average prediction loss on the 'Australia' dataset.

Methods	$K = 1$	$K = 3$	$K = 5$	$K = 10$	$K = 15$
NGFlowVI	0.82	0.65	**0.62**	**0.58**	**0.61**
GFlowVI	0.77	0.67	**0.55**	**0.51**	**0.58**

Table 7: Average prediction loss on the 'Boston' dataset.

Methods	$K = 1$	$K = 3$	$K = 5$	$K = 10$	$K = 15$
NGFlowVI	1.93	1.82	**1.73**	**1.71**	**1.72**
GFlowVI	1.92	1.99	**1.77**	**1.73**	**1.75**

Table 8: Average prediction loss on the 'Concrete' dataset.

Methods	$K = 1$	$K = 3$	$K = 5$	$K = 10$	$K = 15$
NGFlowVI	1.43	**1.26**	**1.27**	**1.25**	**1.26**
GFlowVI	1.67	**1.53**	**1.51**	**1.49**	**1.52**

perform better than SVGD due to their ability to adjust the particle weights, but still perform worse than the others. Possible reasons include the inefficiency of kernels in high-dim problems and suboptimal bandwidth selection for RBF. In addition, NGFlowVI achieves the lowest prediction errors on two datasets 'Boston' and 'Concrete', while GFlowVI achieves the lowest error on 'Australian'. Notably, they outperform the other methods in terms of the prediction loss, indicating the effectiveness of our methods.

Analysis on running time of methods. We compare our methods GFlowVI and NGFlowVI, to SVGD, WVI and NGVI in terms of the computational cost. For GFlowVI, NGFlowVI, WVI and NGVI, we fix the number of components K at 10. For SVGD, we consider the number of particles of 50, 100, 500. We report the average running time (in seconds) required for one epoch of the compared methods in Table 5. For SVGD, the main computational cost arises from the kernel matrix, which requires $O(m^2)$ memory and computation for m particles. In contrast, our methods and WVI, NGVI aim to update parameters of mixture components, with a memory and computational cost of $O(K)$, where K is the number of components. Thus, our methods are particle-efficient compared to SVGD. Furthermore, designing the kernel for SVGD is highly non-trivial, especially for high-dimensional problems.

Analysis on the mixture sizes. To assess the impact of the mixture size (number of components K) on the performance of NGFlowVI and GFlowVI, we conducted an ablation study. We varied the mixture sizes K=1, 3, 5, 10, 15, and evaluated the average prediction losses on three datasets, as shown in Tables 6, 7, and 8. Each method was run for 1000 iterations per mixture size. We see that both NGFlowVI and GFlowVI achieve the highest average losses with K=1, suggesting that a single component may be insufficient to capture the posterior distribution of weights. As the mixture size increases, the losses decrease, demonstrating improved performance. Both methods remain robust with mixture sizes of 10 or more. It suggests to use cross-validation to select the number of components.