Control Variate Sliced Wasserstein Estimators

Khai Nguyen Nhat Ho

The University of Texas at Austin
April 29, 2023

Abstract

The sliced Wasserstein (SW) distances between two probability measures are defined as the expectation of the Wasserstein distance between two one-dimensional projections of the two measures. The randomness comes from a projecting direction that is used to project the two input measures to one dimension. Due to the intractability of the expectation, Monte Carlo integration is performed to estimate the value of the SW distance. Despite having various variants, there has been no prior work that improves the Monte Carlo estimation scheme for the SW distance in terms of controlling its variance. To bridge the literature on variance reduction and the literature on the SW distance, we propose computationally efficient control variates to reduce the variance of the empirical estimation of the SW distance. The key idea is to first find Gaussian approximations of projected one-dimensional measures, then we utilize the closed-form of the Wasserstein-2 distance between two Gaussian distributions to design the control variates. In particular, we propose using a lower bound and an upper bound of the Wasserstein-2 distance between two fitted Gaussians as two computationally efficient control variates. We empirically show that the proposed control variate estimators can help to reduce the variance considerably when comparing measures over images and point-clouds. Finally, we demonstrate the favorable performance of the proposed control variate estimators in gradient flows to interpolate between two point-clouds and in deep generative modeling on standard image datasets, such as CIFAR10 and CelebA.

1 Introduction

Recent machine learning applications have recognized the Wasserstein distance [35, 30], as a universal effective tool. In particular, there are various applications that achieve notable performance by using the Wasserstein distance as a component, for example, (hierarchical) clustering [10], domain adaptation [5, 6], generative modeling [1, 34], and so on. Despite its appealing performance, the Wasserstein distance has two major weaknesses. The first weakness is that it has high computational complexities. As discussed in [30], the time complexity of computing the Wasserstein distance between two discrete measures that have at most n supports is $\mathcal{O}(n^3 \log n)$. Additionally, the space complexity required for the pair-wise transportation cost matrix is $\mathcal{O}(n^2)$. As a result, computing the Wasserstein distance on a large-scale discrete distribution is expensive. The second weakness is that the Wasserstein distance suffers from the curse of dimensionality. More specifically, the sample complexity of the Wasserstein distance is $\mathcal{O}(n^{-1/d})$. Therefore, using the Wasserstein distance in high-dimensional statistical inference may not be stable.

The Wasserstein distance has a famous alternative called the sliced Wasserstein (SW) distance, which is derived from the one-dimensional Wasserstein distance as its base metric. The one-dimensional Wasserstein distance has a closed-form solution, which can be computed in $\mathcal{O}(n \log n)$ time complexity and $\mathcal{O}(n)$ space complexity in the discrete case. Here, 'supports' refer to the discrete probability measures being compared. To apply this closed-form solution in a high-dimension setting, random

projections are employed, transforming two measures into infinite pairs of one-dimensional measures using a projecting function with infinite projecting directions that belong to the unit-hypersphere. The closed-form of the one-dimensional Wasserstein distance is then applied to pairs of one-dimensional measures to obtain projections, and the SW distance is computed by aggregating the values of projections. The most commonly used method of aggregation is averaging, resulting in an expectation of the projected one-dimensional Wasserstein distance with respect to the uniform distribution over projecting directions. Since the space of projecting directions is the unit-hypersphere, the SW distance does not suffer from the curse of dimensionality, and its sample complexity is $\mathcal{O}(n^{-1/2})$. Due to its scalability, the SW distance has been successfully applied in various applications such as point-cloud reconstruction [26, 33], generative models [7, 23], domain adaptation [16], clustering [13], variational inference [37], and many other tasks.

The SW distance is typically estimated using the Monte Carlo integration, as the intractable expectation requires approximation in practice. The number of Monte Carlo samples drawn from the unit-form distribution over the unit-hypersphere is referred to as the number of projections. Increasing the number of projections improves the accuracy and stability of the estimation. However, to the best of our knowledge, there is no prior work on improving the Monte Carlo approximation scheme of the SW distance, especially in terms of variance reduction. In this work, we propose a novel approach that combines the SW distance literature with the variance reduction literature by introducing the very first control variates, which are a technique for reducing the variance of Monte Carlo estimates [29].

Contribution. In summary, our contributions are two-fold:

1. To address the issue of variance in estimating the SW distance using Monte Carlo samples, we propose a novel approach based on control variates. Specifically, we first identify two Gaussian distributions that closely approximate the two projected one-dimensional measures by minimizing Kullback–Leibler divergence. Next, we construct control variates using the closed-form solution of the Wasserstein distance between two Gaussian distributions. We propose two control variates, an upper bound and a lower bound, for the Wasserstein distance between the fitted Gaussian distributions. By using these control variates, we show that the computation is only linear in terms of the number of supports and the number of dimensions, even when dealing with discrete probability measures. This means that the proposed control variates estimators have the same computational complexity as the conventional estimator of the SW distance. Overall, our proposed approach provides a practical and efficient method for estimating the SW distance with reduced variance.

2. We empirically show that the proposed control variate estimators yield considerably smaller variances than the conventional computational estimator of SW distance, using a finite number of projections, when comparing empirical distributions over images and point clouds. Moreover, we illustrate that the computation for control variates is negligible compared to the computation of the one-dimensional Wasserstein distance. Finally, we further demonstrate the favorable performance of the control variate approach in gradient flows between two point clouds, and in learning deep generative models on the CIFAR10 [14] and CelebA [18] datasets. From our experiments, we observe that using the proposed control variate estimators considerably improves performance, while their computation time is approximately the same as that of the conventional estimator.

Organization. The remainder of the paper is organized as follows. First, we review the background on the Wasserstein distance, including its special cases, such as the one-dimensional case and the

Gaussian case, as well as the sliced Wasserstein distance and the Monte Carlo estimation scheme, in Section 2. Next, we discuss the construction of control variates and their resulting estimator of the SW distance in Section 3. We then present experimental results in Section 4 to demonstrate the benefits of the proposed control variate estimators. Finally, we conclude in Section 5 and provide proofs of key results, related works, and additional materials in the Appendices.

Notations. We use the notation $\text{KL}(f, g) = \int_{-\infty}^{\infty} f(x) \log\left(\frac{f(x)}{g(x)}\right) dx$ to denote the Kullback-Leibler divergence between two densities f and g. For any $d \geq 2$, we denote $\mathbb{S}^{d-1} := \theta \in \mathbb{R}^d \mid ||\theta||_2^2 = 1$ as the unit hyper-sphere and $\mathcal{U}(\mathbb{S}^{d-1})$ as its corresponding uniform distribution. We denote $\mathcal{P}(\mathbb{S}^{d-1})$ as the set of all probability measures on $\mathbb{S}^{d-1}$. For $p \geq 1$, $\mathcal{P}_p(\mathbb{R}^d)$ is the set of all probability measures on $\mathbb{R}^d$ that have finite p-moments. For any two sequences an and bn, the notation $a_n = \mathcal{O}(b_n)$ means that $a_n \leq Cb_n$ for all $n \geq 1$, where C is some universal constant. We denote $\theta\sharp\mu$ as the push-forward measure of μ through the function $f : \mathbb{R}^d \to \mathbb{R}$, where $f(x) = \theta^\top x$. We denote P_X as the empirical measure $\frac{1}{m}\sum_{i=1}^m \delta_{x_i}$, where $X := (x_1, \ldots, x_m) \in \mathbb{R}^{dm}$ is a vector."

2 Background

We now review some essential materials for the paper.

Wasserstein distance. Given $p \geq 1$, two probability measures $\mu \in \mathcal{P}_p(\mathbb{R}^d)$ and $\nu \in \mathcal{P}_p(\mathbb{R}^d)$, the Wasserstein distance [36, 30] between μ and ν is : $\text{W}_p^p(\mu, \nu) = \inf_{\pi \in \Pi(\mu,\nu)} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|x - y\|_p^p d\pi(x, y)$, where $\Pi(\mu, \nu)$ is set of all couplings that have marginals are μ and ν respectively. The computational complexity and memory complexity of Wasserstein distance are $\mathcal{O}(n^3 \log n)$ and $\mathcal{O}(n^2)$ in turn when μ and ν have at most n supports. However, there are some special cases where the Wasserstein distance can be computed efficiently.

Special Cases. When $p = 2$, we have Gaussian distributions $\mu := \mathcal{N}(\mathbf{m}_1, \Sigma_1)$ and $\nu := \mathcal{N}(\mathbf{m}_2, \Sigma_2)$, the Wasserstein distance between μ and ν has the following closed-form:

$$\text{W}_2^2(\mu, \nu) = \|\mathbf{m}_1 - \mathbf{m}_2\|_2^2 + \text{Tr}(\Sigma_1 + \Sigma_2 - 2(\Sigma_1^{1/2}\Sigma_2\Sigma_1^{1/2})^{1/2}). \tag{1}$$

When $d = 1$, the Wasserstein distance between $\mu \in \mathcal{P}_p(\mathbb{R})$ and $\nu \in \mathcal{P}_p(\mathbb{R})$ can also be computed with a closed form:

$$\text{W}_p^p(\mu, \nu) = \int_0^1 |F_\mu^{-1}(z) - F_\nu^{-1}(z)|^p dz, \tag{2}$$

where F_μ and F_ν are the cumulative distribution function (CDF) of μ and ν respectively. When μ and ν are one-dimension discrete probability measures that have a most n supports, the computational complexity and memory complexity, in this case, are only $\mathcal{O}(n \log n)$ and $\mathcal{O}(n)$.

Sliced Wasserstein distance. Using random projections, the sliced Wasserstein (SW) distance can exploit the closed-form benefit of Wasserstein distance in one dimension. The definition of sliced Wasserstein distance [3] between two probability measures $\mu \in \mathcal{P}_p(\mathbb{R}^d)$ and $\nu \in \mathcal{P}_p(\mathbb{R}^d)$ is:

$$\text{SW}_p^p(\mu, \nu) = \mathbb{E}_{\theta \sim \mathcal{U}(\mathbb{S}^{d-1})}[\text{W}_p^p(\theta\sharp\mu, \theta\sharp\nu)], \tag{3}$$

where $\mathcal{U}(\mathbb{S}^{d-1})$ is the uniform distribution over the unit-hyper sphere.

Monte Carlo estimation. The expectation in the SW is often intractable, hence, Monte Carlo samples are often used to estimate the expectation:

$$\widehat{\text{SW}}_p^p(\mu,\nu;L) = \frac{1}{L}\sum_{l=1}^{L} \text{W}_p^p(\theta_l\sharp\mu,\theta_l\sharp\nu), \tag{4}$$

where projecting directions $\theta_1,\ldots,\theta_L$ are drawn i.i.d from $\mathcal{U}(\mathbb{S}^{d-1})$. When μ and ν are empirical measures that have at most n supports in d dimension, the time complexity of SW is $\mathcal{O}(Ln\log n + Ldn)$. Here, $Ln\log n$ is for sorting L sets of projected supports and Ldn is for projecting supports to L sets of scalars. Similarly, the space complexity for storing the projecting directions and the projected supports of SW is $\mathcal{O}(Ld + Ln)$. We refer to Algorithm 1 in Appendix B for the detailed algorithm for the SW.

Variance and Error. By using some simple transformations, we can derive the variance of the Monte Carlo approximation of the SW distance as: $\text{Var}[\widehat{\text{SW}}_p^p(\mu,\nu;L)] = \frac{1}{L}\text{Var}[\text{W}_p^p(\theta\sharp\mu,\theta\sharp\nu)]$. We also have the error of the Monte Carlo approximation [21] is:

$$\mathbb{E}\left[|\widehat{\text{SW}}_p^p(\mu,\nu;L) - \text{SW}_p^p(\mu,\nu)|\right] \leq \frac{1}{\sqrt{L}}\text{Var}\left[\text{W}_p^p(\theta\sharp\mu,\theta\sharp\nu)\right]. \tag{5}$$

Here, can see that $\text{Var}[\text{W}_p^p(\theta\sharp\mu,\theta\sharp\nu)$ plays an important role to control the approximation error. Therefore, a natural question arises: "Can we construct a function $Z(\theta;\mu,\nu)$ such that $\mathbb{E}[Z(\theta;\mu,\nu)] = \text{SW}_p^p(\mu,\nu)$ while $\text{Var}[Z(\theta;\mu,\nu)] \leq \text{Var}[\text{W}_p^p(\theta\sharp\mu,\theta\sharp\nu)$?". If we can design such $Z(\theta;\mu,\nu)$, we will have $\mathbb{E}\left[\left|\frac{1}{L}\sum_{l=1}^{L} Z(\theta_l;\mu,\nu) - \text{SW}_p^p(\mu,\nu)\right|\right] \leq \frac{1}{\sqrt{L}}\text{Var}[Z(\theta;\mu,\nu)] \leq \frac{1}{\sqrt{L}}\text{Var}[\text{W}_p^p(\theta\sharp\mu,\theta\sharp\nu)]$ which implies that the estimator $\frac{1}{L}\sum_{l=1}^{L} Z(\theta_l;\mu,\nu)$ is better than $\widehat{\text{SW}}_p^p(\mu,\nu;L)$.

3 Control Variate Sliced Wasserstein Estimators

We first adapt notations from the control variates literature to the SW case in Section 3.1. After that, we discuss how to construct control variates and their computational properties in Section 3.2.

3.1 Control Variate for Sliced Wasserstein Distance

Short Notations. We are interested in approximating the SW which is an expectation with respect to the random variable θ. For convenience, let denote $\text{W}_p^p(\theta\sharp\mu,\theta\sharp\nu)$ as $W(\theta)$, we have:

$$\text{SW}_p^p(\mu,\nu) = \mathbb{E}[W(\theta;\mu,\nu)], \quad \text{and} \quad \widehat{\text{SW}}_p^p(\mu,\nu;L) = \frac{1}{L}\sum_{l=1}^{L} W(\theta_l;\mu,\nu), \tag{6}$$

where $\theta_1,\ldots,\theta_L \overset{i.i.d}{\sim} \sigma_0(\theta) := \mathcal{U}(\mathbb{S}^{d-1})$.

Control Variate. A *control variate* [29] is a random variable $C(\theta)$ such that its expectation is tractable i.e., $\mathbb{E}[C(\theta)] = B$. From the control variate, we can construct a new random variable:

$$Z(\theta;\mu,\nu) = W(\theta;\mu,\nu) - \gamma(C(\theta) - B), \tag{7}$$

where $\gamma \in \mathbb{R}$. Therefore, it is easy to check that $Z(\theta)$ is an *unbiased* estimation of $\mathrm{SW}_p^p(\mu,\nu)$. In particular, $\mathbb{E}[Z(\theta;\mu,\nu)] = \mathbb{E}[W(\theta;\mu,\nu) - \gamma(C(\theta) - B)] = \mathbb{E}[W(\theta)] - \gamma(\mathbb{E}[C(\theta)] - B) = \mathbb{E}[W(\theta;\mu,\nu)] = \mathrm{SW}_p^p(\mu,\nu)$ since $\mathbb{E}[C(\theta)] = B$. Therefore, the Monte Carlo estimation of $\mathbb{E}[Z(\theta;\mu,\nu)]$, i.e., $\frac{1}{L}\sum_{l=1}^{L} Z(\theta_l)$, is also an unbiased estimation of $\mathbb{E}[W(\theta;\mu,\nu)]$. More importantly, the new random variable can have a lower variance than the conventional one. We have:

$$\mathrm{Var}[Z(\theta;\mu,\nu)] = \mathrm{Var}[W(\theta;\mu,\nu)] - 2\gamma\mathrm{Cov}[W(\theta;\mu,\nu), C(\theta)] + \gamma^2\mathrm{Var}[C(\theta)]. \tag{8}$$

Let $f(\gamma) = \mathrm{Var}[W(\theta;\mu,\nu)] - 2\gamma\mathrm{Cov}[W(\theta;\mu,\nu), C(\theta)] + \gamma^2\mathrm{Var}[C(\theta)]$, we have $f'(\gamma) = -2\mathrm{Cov}[W(\theta;\mu,\nu), C(\theta)] + \gamma^2\mathrm{Var}[C(\theta)] + 2\gamma\mathrm{Var}[C(\theta)]$. Setting $f'(\gamma) = 0$, we have $\gamma^\star = \frac{\mathrm{Cov}[W(\theta;\mu,\nu), C(\theta)]}{\mathrm{Var}[C(\theta)]}$ is the optimal choice of γ that can minimize the function $f(\gamma) = \mathrm{Var}[Z(\theta;\mu,\nu)]$. Using the optimal $\gamma^\star$, we have the variance is:

$$\mathrm{Var}[Z(\theta;\mu,\nu)] = \mathrm{Var}[W(\theta;\mu,\nu)]\left(1 - \frac{\mathrm{Cov}[W(\theta;\mu,\nu), C(\theta)]^2}{\mathrm{Var}[W(\theta;\mu,\nu)]\mathrm{Var}[C(\theta)]}\right). \tag{9}$$

Therefore, the variance of $Z(\theta;\mu,\nu)$ is **lower** than $W(\theta;\mu,\nu)$ if the control variate $C(\theta)$ has a correlation with $W(\theta)$. In practice, computing $\gamma^\star$ might be intractable, we can estimate $\gamma^\star$ using Monte Carlo samples. In particular, let $\theta_1, \ldots, \theta_L \sim \sigma_0(\theta)$, we have:

$$\widehat{\mathrm{Var}}[C(\theta); L] = \frac{1}{L}\sum_{l=1}^{L}(C(\theta_l) - B)^2, \quad \widehat{\mathrm{SW}}_p^p(\mu,\nu;L) = \frac{1}{L}\sum_{l=1}^{L} W(\theta_l;\mu,\nu), \tag{10}$$

$$\widehat{\mathrm{Cov}}[W(\theta;\mu,\nu), C(\theta); L] = \frac{1}{L}\sum_{l=1}^{L}(W(\theta_l;\mu,\nu) - \widehat{\mathrm{SW}}_p^p(\mu,\nu;L))(C(\theta_l) - B), \tag{11}$$

$$\widehat{\gamma^\star}_L = \frac{\widehat{\mathrm{Cov}}[W(\theta;\mu,\nu), C(\theta); L]}{\widehat{\mathrm{Var}}[C(\theta); L]}. \tag{12}$$

Control Variate Estimator of the SW distance. Now, we can form the new estimation of $\mathrm{SW}_p^p(\mu,\nu)$:

$$\widehat{\text{CV-SW}}_p^p(\mu,\nu;L) = \widehat{\mathrm{SW}}_p^p(\mu,\nu;L) - \widehat{\gamma^\star}_L\frac{1}{L}\sum_{l=1}^{L}(C(\theta_l) - B), \tag{13}$$

where $\theta_1, \ldots, \theta_L \sim \sigma_0(\theta)$. In this section, we have not yet specified $C(\theta)$. In the next session, we will focus on constructing a computationally efficient $C(\theta)$ that is also effective by conditioning it on two probability measures μ and ν. Specifically, we define $C(\theta) := C(\theta;\mu,\nu)$, which involves conditioning on these two probability measures μ and ν.

3.2 Constructing Control Variates

We now discuss how to design the control variate $C(\theta)$ based on the closed-form of the Wasserstein-2 distance between two Gaussians. We recall that we are interested in the random variable $W(\theta;\mu,\nu) = \mathrm{W}_p^p(\theta\sharp\mu, \theta\sharp\nu)$, hence, it is natural to also construct $C(\theta) := C(\theta;\mu,\nu)$ based on two measures μ,ν. Therefore, it is natural to set $C(\theta;\mu,\nu) = \mathrm{W}_2^2(\mathcal{N}(m_1(\theta;\mu), \sigma_1^2(\theta;\mu)), \mathcal{N}(m_2(\theta;\nu), \sigma_2^2(\theta;\nu)))$.

Gaussian Approximation. The question now is how to specify $\mathcal{N}(m_1(\theta;\mu),\sigma_1^2(\theta;\mu))$ and $\mathcal{N}(m_2(\theta;\nu),\sigma_2^2(\theta;\nu))$. In particular, we perform the following optimization problems:

$$m_1(\theta;\mu),\sigma_1^2(\theta;\mu) = \text{argmin}_{m_1,\sigma_1^2}\text{KL}(\theta\sharp\mu,\mathcal{N}(m_1,\sigma_1^2)), \tag{14}$$

$$m_2(\theta;\nu),\sigma_2^2(\theta;\nu) = \text{argmin}_{m_2,\sigma_2^2}\text{KL}(\theta\sharp\nu,\mathcal{N}(m_2,\sigma_2^2)), \tag{15}$$

where KL denotes the Kullback–Leibler divergence. We first consider the discrete case, namely, $\mu = \sum_{i=1}^n \alpha_i\delta_{x_i}$ $(\sum_{i=1}^n \alpha_i = 1)$ and $\nu = \sum_{i=1}^m \beta_i\delta_{y_i}$ $(\sum_{i=1}^m \beta_i = 1)$.

Proposition 1. *Let μ and ν be two discrete probability measures i.e., $\mu = \sum_{i=1}^n \alpha_i\delta_{x_i}$ $(\sum_{i=1}^n \alpha_i = 1)$ and $\nu = \sum_{i=1}^m \beta_i\delta_{y_i}$ $(\sum_{i=1}^m \beta_i = 1)$, we have:*

$$m_1(\theta;\mu) = \sum_{i=1}^n \alpha_i\theta^\top x_i, \quad \sigma_1^2(\theta;\mu) = \sum_{i=1}^n \alpha_i\left(\theta^\top x_i - m_1(\theta;\mu)\right)^2 \tag{16}$$

$$m_2(\theta;\nu) = \sum_{i=1}^m \beta_i\theta^\top y_i, \quad \sigma_2^2(\theta;\nu) = \sum_{i=1}^m \beta_i\left(\theta^\top y_i - m_2(\theta;\nu)\right)^2, \tag{17}$$

are solution of the problems in equation 14- 15.

We refer to Appendix A.1 for the detailed proof of Proposition 1. When μ and ν are continuous, we can use their empirical approximation $\mu_n = \frac{1}{n}\sum_{i=1}^n \delta_{X_i}$ and $\nu_n = \frac{1}{n}\sum_{i=1}^n \delta_{Y_i}$ as proxies (where $X_1,\ldots,X_n \overset{i.i.d}{\sim} \mu$ and $Y_1,\ldots,Y_n \overset{i.i.d}{\sim} \nu$) or use the Laplace approximation to obtain two approximated Gaussians for $\theta\sharp\mu$ and $\theta\sharp\nu$ when the push forward densities are tractable.

Constructing Control Variates. From the closed-form of the Wasserstein-2 distance between two Gaussians in equation 1, we have:

$$\begin{aligned}&\text{W}_2^2(\mathcal{N}(m_1(\theta;\mu),\sigma_1^2(\theta;\mu)),\mathcal{N}(m_2(\theta;\nu),\sigma_2^2(\theta;\nu))) = (m_1(\theta;\mu) - m_2(\theta;\nu))^2\\ &+ (\sigma_1(\theta;\mu) - \sigma_2(\theta;\nu))^2 = (m_1(\theta;\mu) - m_2(\theta;\nu))^2 + \sigma_1(\theta;\mu)^2 + \sigma_2(\theta;\nu)^2 - 2\sigma_1(\theta;\mu)\sigma_2(\theta;\nu).\end{aligned}$$

Now, we calculate $\mathbb{E}[\text{W}_2^2(\mathcal{N}(m_1(\theta;\mu),\sigma_1^2(\theta;\mu)),\mathcal{N}(m_2(\theta;\nu),\sigma_2^2(\theta;\nu)))]$ as the requirement of a control variate. Using the linearity of expectation, we could calculate the expectation of each term:

$$\begin{aligned}&\mathbb{E}[(m_1(\theta;\mu) - m_2(\theta;\nu))^2] = \mathbb{E}\left[\left(\sum_{i=1}^n \alpha_i\theta^\top x_i - \sum_{i=1}^m \beta_i\theta^\top y_i\right)^2\right]\\ &= \left(\sum_{i=1}^n \alpha_i x_i - \sum_{i=1}^m \beta_i y_i\right)^\top \mathbb{E}\left[\theta\theta^\top\right]\left(\sum_{i=1}^n \alpha_i x_i - \sum_{i=1}^m \beta_i y_i\right) = \frac{1}{d}\left\|\sum_{i=1}^n \alpha_i x_i - \sum_{i=1}^m \beta_i y_i\right\|^2,\end{aligned}$$

where we use the result of $\mathbb{E}_{\theta\sim\mathcal{U}(\mathbb{S}^{d-1})}[\theta\theta^\top] = \frac{1}{d}$. Next, we have:

$$\begin{aligned}&\mathbb{E}\left[\sigma_1^2(\theta;\mu)\right] = \mathbb{E}\left[\sum_{i=1}^n \alpha_i\left(\theta^\top x_i - \sum_{i'=1}^n \alpha_{i'}\theta^\top x_{i'}\right)^2\right]\\ &= \sum_{i=1}^n \alpha_i\left(x_i - \sum_{i'=1}^n \alpha_{i'}x_{i'}\right)^\top \mathbb{E}\left[\theta\theta^\top\right]\left(x_i - \sum_{i'=1}^n \alpha_{i'}x_{i'}\right) = \frac{1}{d}\sum_{i=1}^n \alpha_i\left\|x_i - \sum_{i'=1}^n \alpha_{i'}x_{i'}\right\|^2.\end{aligned}$$

Similarly, we have $\mathbb{E}\left[\sigma_2^2(\theta;\nu)\right] = \frac{1}{d}\sum_{i=1}^m \beta_i \left\|y_i - \sum_{i'=1}^m \beta_{i'} y_{i'}\right\|^2$. Unfortunately, we cannot compute the expectation for the last term $\mathbb{E}[\sigma_1(\theta;\mu)\sigma_2(\theta;\nu)]$. Therefore, we construct control variates which are a lower-bound and an upper-bound of $\mathrm{W}_2^2(\mathcal{N}(m_1(\theta;\mu),\sigma_1^2(\theta;\mu)),\mathcal{N}(m_2(\theta;\nu),\sigma_2^2(\theta;\nu)))$. In particular, we have the following control variates:

$$C_{low}(\theta;\mu,\nu) = (m_1(\theta;\mu) - m_2(\theta;\nu))^2 = \left(\sum_{i=1}^n \alpha_i \theta^\top x_i - \sum_{i=1}^m \beta_i \theta^\top y_i\right)^2, \tag{18}$$

$$C_{up}(\theta;\mu,\nu) = (m_1(\theta;\mu) - m_2(\theta;\nu))^2 + \sigma_1^2(\theta;\mu) + \sigma_2^2(\theta;\nu) = \left(\sum_{i=1}^n \alpha_i \theta^\top x_i - \sum_{i=1}^m \beta_i \theta^\top y_i\right)^2 + \sum_{i=1}^n \alpha_i \left(\theta^\top x_i - \sum_{i=1}^n \alpha_i \theta^\top x_i\right)^2 + \sum_{i=1}^m \beta_i \left(\theta^\top y_i - \sum_{i=1}^m \beta_i \theta^\top y_i\right)^2. \tag{19}$$

We now discuss some properties of the proposed control variates.

Proposition 2. *We have the following relationship:*

$$C_{low}(\theta;\mu,\nu) \leq W_2^2(\mathcal{N}(m_1(\theta;\mu),\sigma_1^2(\theta;\mu)),\mathcal{N}(m_2(\theta;\nu),\sigma_2^2(\theta;\nu))) \leq C_{up}(\theta;\mu,\nu). \tag{20}$$

The proof of Proposition 2 follows directly from the construction of the control variates. For completeness, we provide the proof in Appendix A.2.

Proposition 3. *Let $\mu = \sum_{i=1}^n \alpha_i \delta_{x_i}$ and $\nu = \sum_{i=1}^m \beta_i \delta_{y_i}$, using the control variate $C_{low}(\theta;\mu,\nu)$ is equivalent to using the following control variate:*

$$W_2^2(\theta\sharp\mathcal{N}(\mathbf{m_1}(\mu),\sigma_1^2(\mu)\mathbf{I}),\theta\sharp\mathcal{N}(\mathbf{m_2}(\nu),\sigma_2^2(\nu)\mathbf{I})), \tag{21}$$

where we obtain $\mathbf{m_1}(\mu),\sigma_1^2(\mu) = argmin_{\mathbf{m_1},\sigma_1^2} KL(\mu,\mathcal{N}(\mathbf{m_1},\sigma_1^2))$ and $\mathbf{m_2}(\nu),\sigma_2^2(\nu) = argmin_{\mathbf{m_2},\sigma_2^2} KL(\nu,\mathcal{N}(\mathbf{m_2},\sigma_2^2))$.

The proof for The Proposition 3 is given in Appendix A.3. The proposition means that using the control variate $C_{low}(\theta;\mu,\nu)$ is equivalent to using the Wasserstein-2 distance between two projected *location-scale multivariate* Gaussians with these location-scale multivariate Gaussians are the approximation of the two original probability measures μ and ν on the original space.

Control Variate Estimators of the SW distance. Using the above control variates, we have:

$$Z_{low}(\theta;\mu,\nu) = W(\theta;\mu,\nu) - \frac{\mathrm{Cov}[W(\theta;\mu,\nu), C_{low}(\theta;\mu,\nu)]}{\mathrm{Var}[C_{low}(\theta;\mu,\nu)]}(C_{low}(\theta;\mu,\nu) - B_{low}(\mu,\nu)), \tag{22}$$

$$Z_{up}(\theta;\mu,\nu) = W(\theta;\mu,\nu) - \frac{\mathrm{Cov}[W(\theta;\mu,\nu), C_{up}(\theta;\mu,\nu)]}{\mathrm{Var}[C_{up}(\theta;\mu,\nu)]}(C_{up}(\theta;\mu,\nu) - B_{up}(\mu,\nu)) \tag{23}$$

where the expected values of the control variates: $B_{low}(\mu,\nu) = \frac{1}{d}\left\|\sum_{i=1}^n \alpha_i x_i - \sum_{i=1}^m \beta_i y_i\right\|^2$ and $B_{up}(\mu,\nu) = \frac{1}{d}\left\|\sum_{i=1}^n \alpha_i x_i - \sum_{i=1}^m \beta_i y_i\right\|^2 + \frac{1}{d}\sum_{i=1}^n \alpha_i \left\|x_i - \sum_{i=1}^n \alpha_i x_i\right\|^2 + \frac{1}{d}\sum_{i=1}^m \beta_i \left\|y_i - \sum_{i=1}^m \beta_i y_i\right\|^2$. Hence, we have the following two new control variate estimators of the SW distance:

$$\widehat{\text{LCV-SW}}_p^p(\mu,\nu;L) = \widehat{\text{SW}}_p^p(\mu,\nu;L) - \widehat{\gamma^\star}_{L,low}(\mu,\nu)\frac{1}{L}\sum_{l=1}^L \left(C_{low}(\theta_l;\mu,\nu) - B_{low}(\mu,\nu)\right), \tag{24}$$

$$\widehat{\text{UCV-SW}}_p^p(\mu,\nu;L) = \widehat{\text{SW}}_p^p(\mu,\nu;L) - \widehat{\gamma^\star}_{L,up}(\mu,\nu)\frac{1}{L}\sum_{l=1}^L \left(C_{up}(\theta_l;\mu,\nu) - B_{up}(\mu,\nu)\right), \tag{25}$$

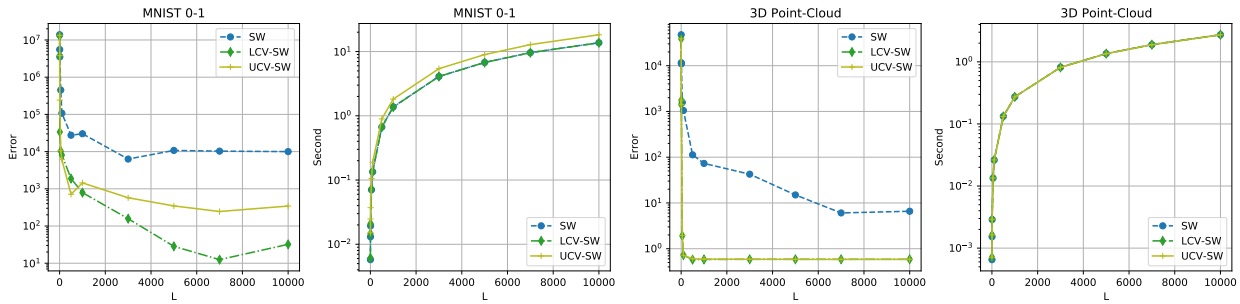

Figure 1: The empirical errors of the conventional estimator (SW) and the control variate estimators (LCV-SW, UCV-SW) when comparing empirical distributions over MNIST images and point-clouds.

where Monte Carlo samples $\theta_1, \ldots, \theta_L \overset{i.i.d}{\sim} \sigma_0(\theta) = \mathcal{U}(\mathbb{S}^{d-1})$, estimators $\widehat{\gamma^\star}_{L,low}(\mu,\nu) = \frac{\widehat{\text{Cov}}[W(\theta;\mu,\nu),C_{low}(\theta;\mu,\nu);L]}{\widehat{\text{Var}}[C_{low}(\theta;\mu,\nu);L]}$, and $\widehat{\gamma^\star}_{L,up}(\mu,\nu) = \frac{\widehat{\text{Cov}}[W(\theta;\mu,\nu),C_{up}(\theta;\mu,\nu);L]}{\widehat{\text{Var}}[C_{up}(\theta;\mu,\nu);L]}$ with the Monte Carlo estimators of the covariance and the variance are defined in equation 10- 11. We refer to Algorithm 2- 3 for the detailed algorithms for the control variate estimators in Appendix B.

Computational Complexities. When dealing with two discrete probability measures μ and ν that have at most n supports, projecting the supports of the two measures to L sets of projected supports costs $\mathcal{O}(Ldn)$ in time complexity. After that, fitting one-dimensional Gaussians also costs $\mathcal{O}(Ldn)$. Computing the control variate $C_{low}(\theta;\mu,\nu)$ and its expected value $B_{low}(\mu,\nu)$ costs $\mathcal{O}(dn)$. Similarly, computing the control variate $C_{up}(\theta;\mu,\nu)$ and its expected value B_{up} also costs $\mathcal{O}(dn)$. Overall, the time complexity of $\widehat{\text{LCV-SW}}_p^p(\mu,\nu;L)$ and $\widehat{\text{UCV-SW}}_p^p(\mu,\nu;L)$ is the same as the conventional SW, which is $\mathcal{O}(Ln\log n + Ldn)$. Similarly, their space complexities are also $\mathcal{O}(Ld + Ln)$.

Gaussian Approximation of the sliced Wasserstein distance. In a recent work [22], the closed-form of Wasserstein between Gaussian distributions is utilized to approximate the value of the SW distance. The key idea is based on the concentration of random projections in high-dimension, i.e., they are approximately Gaussian. However, the resulting deterministic approximation is not very accurate since it is only based on the first two moments of two original measures. As a result, the proposed approximation can only work well when two measures have weak dimensional dependence. In contrast, we use the closed-form of Wasserstein between Gaussian distributions to design control variates, which can reduce the variance of the Monte Carlo estimator. Our estimators are still stochastic, converge to the true value of the SW when increasing the number of projections, and work well with probability measures that have arbitrary structures and dimensions.

Beyond uniform slicing distribution and related works. In the SW distance, the projecting direction follows the uniform distribution over the unit-hypersphere $\sigma_0(\theta) = \mathcal{U}(\mathbb{S}^{d-1})$. In some cases, changing the slicing distribution might benefit downstream applications. For example, the distributional sliced Wasserstein distance [25, 27] can be written as $\text{DSW}_p^p(\mu,\nu) = \mathbb{E}_{\theta\sim\sigma(\theta)}\left[\text{W}_p^p(\theta\sharp\mu,\theta\sharp\nu)\right]$, where $\sigma(\theta)$ is a distribution over the unit hypersphere. We can directly apply the proposed control variates to the DSW as long as we can calculate $\mathbb{E}_{\theta\sim\sigma(\theta)}[\theta\theta^\top]$. If it is not the case, we can still use the proposed control variates by approximating the DSW via importance sampling. In particular, we can rewrite $\text{DSW}_p^p(\mu,\nu) = \mathbb{E}_{\theta\sim\sigma_0(\theta)}\left[\text{W}_p^p(\theta\sharp\mu,\theta\sharp\nu)\frac{\sigma(\theta)}{\sigma_0(\theta)}\right]$, then use control variates. We discuss other related works including how we can apply the control variates to other variants of the SW in Appendix C.

Table 1: Summary of Wasserstein-2 scores (multiplied by 10^4), computational time in second (s) to reach step 500 of different sliced Wasserstein variants in gradient flows from three different runs.

Distances	Step 3000 ($W_2 \downarrow$)	Step 4000 ($W_2 \downarrow$)	Step 5000 ($W_2 \downarrow$)	Step 6000($W_2 \downarrow$)	Step 8000 ($W_2 \downarrow$)	Time (s $\downarrow$)
SW L=10	305.5196 ± 0.4921	137.8767 ± 0.3631	36.2454 ± 0.1387	0.1164 ± 0.0022	$2.1e-5 \pm 1.0e-5$	$\mathbf{26.13 \pm 0.04}$
LCV-SW L=10	$\mathbf{303.1001 \pm 0.1787}$	$\mathbf{136.0129 \pm 0.0923}$	35.0929 ± 0.1459	0.0538 ± 0.0047	$1.5e-5 \pm 0.2e-5$	27.22 ± 0.24
UCV-SW L=10	303.1017 ± 0.1783	136.0139 ± 0.0921	$\mathbf{35.0916 \pm 0.1444}$	$\mathbf{0.0535 \pm 0.0065}$	$\mathbf{1.4e-5 \pm 0.2e-5}$	29.48 ± 0.23
SW L=100	300.7326 ± 0.2375	134.1498 ± 0.3146	33.9253 ± 0.1349	0.0183 ± 0.0011	$1.6e-5 \pm 0.2e-5$	$\mathbf{220.20 \pm 0.21}$
LCV-SW L=100	$\mathbf{300.3924 \pm 0.0053}$	133.6243 ± 0.0065	33.5102 ± 0.0031	$\mathbf{0.0134 \pm 8.7e-5}$	$\mathbf{1.4e-5 \pm 0.1e-5}$	221.75 ± 0.35
UCV-SW L=100	300.3927 ± 0.0050	$\mathbf{133.6242 \pm 0.0065}$	$\mathbf{33.5088 \pm 0.0028}$	0.0136 ± 0.0003	$1.4e-5 \pm 0.2e-5$	233.71 ± 0.06

Gradient estimators. In some cases, we might be interested in obtaining the gradient $\nabla_\phi \mathrm{SW}_p^p(\mu, \nu_\phi) = \nabla_\phi \mathbb{E}[\mathrm{W}_p^p(\theta\sharp\mu, \theta\sharp\nu_\phi)]$ e.g., statistical inference. Since the expectation in the SW does not depend on ϕ, we can use the Leibniz rule to exchange differentiation and expectation, namely, $\nabla_\phi \mathbb{E}[\mathrm{W}_p^p(\theta\sharp\mu, \theta\sharp\nu_\phi)] = \mathbb{E}[\nabla_\phi \mathrm{W}_p^p(\theta\sharp\mu, \theta\sharp\nu_\phi)]$. After that, we can use $\nabla_\phi C(\theta; \mu, \nu_\phi)$ ($\nabla_\phi C_{low}(\theta; \mu, \nu_\phi)$ or $\nabla_\phi C_{up}(\theta; \mu, \nu_\phi)$) as the control variate. However, estimating the optimal $\gamma_i^\star = \frac{\mathrm{Cov}[\nabla_{\phi_i} \mathrm{W}_p^p(\theta\sharp\mu, \theta\sharp\nu_\phi), \nabla_{\phi_i} C(\theta;\mu,\nu_\phi)]}{\mathrm{Var}[\nabla_{\phi_i} C(\theta;\mu,\nu_\phi)]}$ (for $i = 1, \ldots, d'$ with d' is the number of dimensions of parameters ϕ) is computationally expensive and depends significantly on the specific settings of ν_ϕ. Therefore, we utilize a more computationally efficient estimator of the gradient i.e., $\mathbb{E}[\nabla_\phi Z(\theta; \mu, \nu_\phi)]$ for $Z(\theta; \mu, \nu_\phi)$ is $Z_{low}(\theta; \mu, \nu_\phi)$ or $Z_{up}(\theta; \mu, \nu_\phi)$ in equation 22- 23. After that, Monte Carlo samples are used to approximate expectations to obtain a stochastic gradient.

4 Experiments

Our experiments aim to show that using control variates can benefit applications of the sliced Wasserstein distance. In particular, we show that the proposed control variate estimators have lower variance in practice when comparing probability measures over images and point-clouds in Section 4.1. After that, we show that using the proposed control variate estimators can help to drive a gradient flow to converge faster to a target probability measure from a source probability measure while retaining approximately the same computation in Section 4.2. Finally, we show the proposed control variate estimators can also improve training deep generative models on standard benchmark images dataset such as CIFAR10, and CelebA in Section 4.3. Due to the space constraint, additional experiments and detailed experimental settings are deferred to Appendix D.

4.1 Comparing empirical probability measures over images and point-clouds

Settings. We aim to first test how well the proposed control variates can reduce the variance in practice. To do this, we use conventional Monte Carlo estimation for the SW with a very large L value of 50000 and treat it as the population value. Next, we vary L in the set $\{2, 5, 10, 50, 100, 500, 1000, 5000, 7000, 10000\}$ and compute the corresponding estimates of the SW with both the conventional estimator and control variate estimators. Finally, we calculate the squared difference between the estimators and the estimated population as the estimated error. We apply this process to compare the empirical distributions over images of digit 0 with the empirical distribution images of digit 1 in MNIST [15] (784 dimensions, with about 6000 supports). Similarly, we compare two empirical distributions over two randomly chosen point-clouds in the ShapeNet Core-55 dataset [4] (3 dimensions, 2048 supports). We also estimate the variance of the estimators via Monte Carlo samples with $L = 50000$ in Table 3 in Appendix D.1.

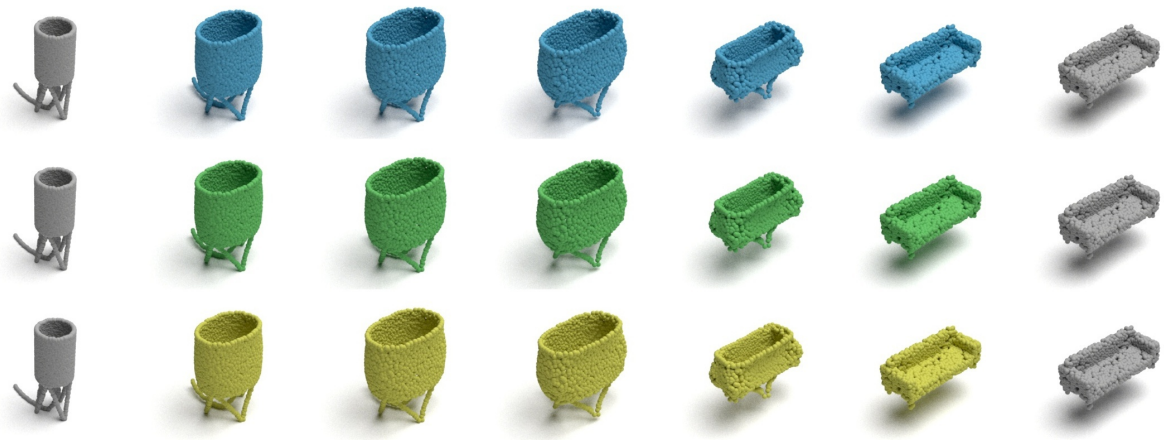

Figure 2: Point-cloud gradient flows from SW, LCV-SW, and UCV-SW respectively.

Results. We show the estimated errors and the computational time which are averaged from 5 different runs in Figure 1. From the figure, we observe that control variate estimators reduce considerably the error in both cases. On MNIST, the lower bound control variate estimator (LCV-SW) gives a lower error than the upper bound control variate estimator (UCV-SW) while being slightly faster. The computational time of the LCV-SW has nearly identical computational time as the conventional estimator (SW). In the lower dimension i.e., the point-cloud case, two control variates give the same quality of reducing error and the same computational time. In this case, the computational time of both two control variates estimators are the same as the conventional estimator. From Table 3 in Appendix D.1, we observe that the variance of the control variate estimators is lower than the conventional estimator considerably, especially the LCV-SW. We refer to Appendix D.1 for experiments on different pairs of digits and point-clouds which show the same phenomenon.

4.2 Point Cloud Gradient Flows

Settings. We model a distribution $\mu(t)$ flowing with time t along the sliced Wasserstein gradient flow $\mu(t) \to \text{SW}_p(\mu(t), \nu)$, which drives it towards a target distribution ν [32]. Here, we set $\nu = \frac{1}{n}\sum_{i=1}^n \delta_{Y_i}$ as a fixed empirical target distribution and the model distribution $\mu(t) = \frac{1}{n}\sum_{i=1}^n \delta_{X_i(t)}$, where the time-varying point cloud $X(t) = (X_i(t))_{i=1}^n \in \left(\mathbb{R}^d\right)^n$. Starting at time $t = 0$, we integrate the ordinary differential equation $\dot{X}(t) = -n\nabla_{X(t)}\left[\text{SW}_p\left(\frac{1}{n}\sum_{i=1}^n \delta_{X_i(t)}, \nu\right)\right]$ for each iteration. We choose $\mu(0)$ and ν are two random point-cloud shapes in ShapeNet Core-55 dataset [4] which were used in Section 4.1. After that, we use different estimators of the SW i.e., $\widehat{\text{SW}}_p$, $\widehat{\text{LCV-SW}}_p$, and $\widehat{\text{UCV-SW}}_p$, as the replacement of the for SW_p to estimate the gradient including the proposed control variate estimators and the conventional estimator. Finally, we use $p = 2$, and the Euler scheme with 8000 iterations and step size 0.01 to solve the flows.

Results. We use the Wasserstein-2 [8] distance as a neutral metric to evaluate how close the model distribution $\mu(t)$ is to the target distribution ν. We show the results for the conventional Monte Carlo estimator (denoted as SW), the control variate C_{low} estimator (denoted as LCV-SW), and the control variate C_{up} estimator (denoted as UCV-SW), with the number of projections $L = 10, 100$ in Table 1. In the table, we also report the time in seconds that estimators need to reach the last step.

SW L=10 LCV-SW L=10 UCV-SW L=10

Figure 3: Random generated images of distances on CIFAR10 and CelebA.

Table 2: Summary of FID and IS scores from three different runs on CIFAR10 (32x32), and CelebA (64x64).

Method	CIFAR10 (32x32)		CelebA (64x64)	Method	CIFAR10 (32x32)		CelebA (64x64)
	FID (↓)	IS (↑)	FID (↓)		FID (↓)	IS (↑)	FID (↓)
SW L=10	21.72 ± 0.37	7.54 ± 0.06	11.14 ± 1.12	SW L=1000	14.07 ± 0.97	8.07 ± 0.06	10.05 ± 0.40
LCV-SW L=10	$\mathbf{18.67 \pm 0.39}$	$\mathbf{7.74 \pm 0.04}$	$\mathbf{10.17 \pm 0.73}$	LCV-SW L=1000	13.58 ± 0.12	$\mathbf{8.23 \pm 0.02}$	9.78 ± 0.47
UCV-SW L=10	21.71 ± 0.36	7.57 ± 0.01	11.00 ± 0.04	UCV-SW L=1000	$\mathbf{13.21 \pm 0.49}$	8.21 ± 0.09	$\mathbf{9.51 \pm 0.30}$

We observe the same phenomenon on a different pair of point-clouds in Table 4 and in Figure 5 in Appendix D.2.

4.3 Deep Generative Modeling

Settings. We conduct deep generative modeling on CIFAR10 (with image size 32x32) [14], *non-croppeed* CelebA (with image size 64x64) [18] with the SNGAN architecture [19]. For the framework, we follow the sliced Wasserstein generator [7, 23] which is described in detail in Appendix D.3. The main evaluation metrics are FID score [9] and Inception score (IS) [31]. On CelebA, we do not report the IS since it poorly captures the perceptual quality of face images [9]. The detailed settings about architectures, hyper-parameters, and evaluation of FID and IS are provided in Appendix D.3.

Results. We train generators with the conventional estimator (SW), and the proposed control variates estimators (LCV-SW and UCV-SW) with the number of projections $L = 10, 1000$. We report FID scores and IS scores in Table 2. From the table, we observe that the LCV-SW gives the

best generative models on both CIFAR10 and CelebA when $L = 10$. In particular, the LCV-SW can reduce about 14% FID score on CIFAR10, and about 10% FID score on CelebA. Also, the LCV-SW increases the IS score by about 2.6%. For the UCV-SW, it can enhance the performance slightly compared with the SW when $L = 10$. It is worth noting that, the computational times of estimators are approximately the same since the main computation in the application is for training neural networks. Therefore, we can see the benefit of reducing the variance by using control variates here. Increasing L to 1000, despite the gap being closer, the control variate estimators still give better scores than the conventional estimator. In greater detail, the LCV-SW can improve about 3.5% FID score on CIFAR10, about 2% IS score on CIFAR10, and about 2.7% FID score on CelebA compared to the SW. In this setting of L, the UCV-SW gives the best FID scores on both CIFAR10 and CelebA. Concretely, it helps to decrease the FID score by about 6.1% on CIFAR10 and about 5.4% on CelebA. From the result, the UCV-SW seems to need more projections than the LCV-SW to approximate well the optimal control variate coefficient ($\gamma^\star$) since the corresponding control variate of the UCV-SW has two more random terms. For the qualitative result, we show randomly generated images from trained models in Figure 3 for $L = 10$ and in Figure 6 for $L = 1000$ in Appendix D.3. Overall, we obverse that generated images are visually consistent to the FID scores and IS scores in Table 2.

5 Conclusion

We have presented a method for reducing the variance of Monte Carlo estimation of the sliced Wasserstein (SW) distance using control variates. Moreover, we propose two novel control variates that serve as lower and upper bounds for the Wasserstein distance between two Gaussian approximations of two measures. By using the closed-form solution of the Wasserstein distance between two Gaussians and the closed-form of Gaussian fitting of discrete measures via Kullback Leibler divergence, we demonstrate that the proposed control variates can be computed in linear time. Consequently, the control variate estimators have the same computational complexity as the conventional estimator of SW distance. On the experimental side, we demonstrate that the proposed control variate estimators have smaller variances than the conventional estimator when comparing probability measures over images and point clouds. Finally, we show that using the proposed control variate estimators can lead to improved performance of point-cloud SW gradient flows and generative models.

References

[1] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223, 2017. (Cited on page 1.)

[2] C. Bonet, P. Berg, N. Courty, F. Septier, L. Drumetz, and M. T. Pham. Spherical sliced-wasserstein. In *The Eleventh International Conference on Learning Representations*, 2023. (Cited on page 20.)

[3] N. Bonneel, J. Rabin, G. Peyré, and H. Pfister. Sliced and Radon Wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 1(51):22–45, 2015. (Cited on page 3.)

[4] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. (Cited on pages 9 and 10.)

[5] N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonjy. Optimal transport for domain adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 39(9):1853–1865, 2016. (Cited on page 1.)

[6] B. B. Damodaran, B. Kellenberger, R. Flamary, D. Tuia, and N. Courty. Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 447–463, 2018. (Cited on page 1.)

[7] I. Deshpande, Z. Zhang, and A. G. Schwing. Generative modeling using the sliced Wasserstein distance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3483–3491, 2018. (Cited on pages 2 and 11.)

[8] R. Flamary, N. Courty, A. Gramfort, M. Z. Alaya, A. Boisbunon, S. Chambon, L. Chapel, A. Corenflos, K. Fatras, N. Fournier, L. Gautheron, N. T. Gayraud, H. Janati, A. Rakotomamonjy, I. Redko, A. Rolet, A. Schutz, V. Seguy, D. J. Sutherland, R. Tavenard, A. Tong, and T. Vayer. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021. (Cited on page 10.)

[9] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017. (Cited on page 11.)

[10] N. Ho, X. Nguyen, M. Yurochkin, H. H. Bui, V. Huynh, and D. Phung. Multilevel clustering via Wasserstein means. In *International Conference on Machine Learning*, pages 1501–1509, 2017. (Cited on page 1.)

[11] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. (Cited on page 23.)

[12] S. Kolouri, K. Nadjahi, U. Simsekli, R. Badeau, and G. Rohde. Generalized sliced Wasserstein distances. In *Advances in Neural Information Processing Systems*, pages 261–272, 2019. (Cited on page 20.)

[13] S. Kolouri, G. K. Rohde, and H. Hoffmann. Sliced Wasserstein distance for learning Gaussian mixture models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3427–3436, 2018. (Cited on page 2.)

[14] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009. (Cited on pages 2 and 11.)

[15] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. (Cited on page 9.)

[16] C.-Y. Lee, T. Batra, M. H. Baig, and D. Ulbricht. Sliced Wasserstein discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10285–10295, 2019. (Cited on pages 2 and 20.)

[17] T. Li, C. Meng, J. Yu, and H. Xu. Hilbert curve projection distance for distribution comparison. *arXiv preprint arXiv:2205.15059*, 2022. (Cited on page 20.)

[18] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015. (Cited on pages 2 and 11.)

[19] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018. (Cited on page 11.)

[20] K. Nadjahi, V. De Bortoli, A. Durmus, R. Badeau, and U. Şimşekli. Approximate Bayesian computation with the sliced-Wasserstein distance. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5470–5474. IEEE, 2020. (Cited on page 20.)

[21] K. Nadjahi, A. Durmus, L. Chizat, S. Kolouri, S. Shahrampour, and U. Simsekli. Statistical and topological properties of sliced probability divergences. *Advances in Neural Information Processing Systems*, 33:20802–20812, 2020. (Cited on page 4.)

[22] K. Nadjahi, A. Durmus, P. E. Jacob, R. Badeau, and U. Simsekli. Fast approximation of the sliced-Wasserstein distance using concentration of random projections. *Advances in Neural Information Processing Systems*, 34, 2021. (Cited on page 8.)

[23] K. Nguyen and N. Ho. Amortized projection optimization for sliced Wasserstein generative models. *Advances in Neural Information Processing Systems*, 2022. (Cited on pages 2 and 11.)

[24] K. Nguyen and N. Ho. Revisiting sliced Wasserstein on images: From vectorization to convolution. *Advances in Neural Information Processing Systems*, 2022. (Cited on page 20.)

[25] K. Nguyen, N. Ho, T. Pham, and H. Bui. Distributional sliced-Wasserstein and applications to generative modeling. In *International Conference on Learning Representations*, 2021. (Cited on page 8.)

[26] K. Nguyen, D. Nguyen, and N. Ho. Self-attention amortized distributional projection optimization for sliced wasserstein point-cloud reconstruction. *Proceedings of the 40th International Conference on Machine Learning*, 2023. (Cited on page 2.)

[27] K. Nguyen, S. Nguyen, N. Ho, T. Pham, and H. Bui. Improving relational regularized autoencoders with spherical sliced fused Gromov-Wasserstein. In *International Conference on Learning Representations*, 2021. (Cited on page 8.)

[28] K. Nguyen, T. Ren, H. Nguyen, L. Rout, T. M. Nguyen, and N. Ho. Hierarchical sliced wasserstein distance. In *The Eleventh International Conference on Learning Representations*, 2023. (Cited on page 20.)

[29] A. B. Owen. Monte carlo theory, methods and examples. 2013. (Cited on pages 2 and 4.)

[30] G. Peyré and M. Cuturi. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019. (Cited on pages 1 and 3.)

[31] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training GANs. *Advances in Neural Information Processing Systems*, 29, 2016. (Cited on page 11.)

[32] F. Santambrogio. Optimal transport for applied mathematicians. *Birkäuser, NY*, 55(58-63):94, 2015. (Cited on page 10.)

[33] A. Savkin, Y. Wang, S. Wirkert, N. Navab, and F. Tombari. Lidar upsampling with sliced wasserstein distance. *IEEE Robotics and Automation Letters*, 8(1):392–399, 2022. (Cited on page 2.)

[34] I. Tolstikhin, O. Bousquet, S. Gelly, and B. Schoelkopf. Wasserstein auto-encoders. In *International Conference on Learning Representations*, 2018. (Cited on page 1.)

[35] C. Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008. (Cited on page 1.)

[36] C. Villani. *Optimal transport: Old and New*. Springer, 2008. (Cited on page 3.)

[37] M. Yi and S. Liu. Sliced Wasserstein variational inference. In *Fourth Symposium on Advances in Approximate Bayesian Inference*, 2021. (Cited on page 2.)

Supplement to "Control Variate Sliced Wasserstein Estimators"

We first provide the skipped proofs in Appendix A. Additionally, we present the detailed algorithms for the conventional estimator of the sliced Wasserstein distance and the control variate estimators in Appendix B. Sliced Wasserstein variants are discussed in Appendix C, and we also discuss how to use control variates in those variants in the same appendix. In Appendix D, we provide additional experiments and their detailed experimental settings. Finally, we report the computational infrastructure in Appendix E.

A Proofs

A.1 Proof of Proposition 1

We have μ and ν are two empirical measures i.e., $\mu = \sum_{i=1}^n \alpha_i \delta_{x_i}$ ($\sum_{i=1}^n \alpha_i = 1$) and $\nu = \sum_{i=1}^m \beta_i \delta_{y_i}$ ($\sum_{i=1}^m \beta_i = 1$), we have their projected measures $\theta\sharp\mu = \sum_{i=1}^n \alpha_i \delta_{\theta^\top x_i}$ and $\theta\sharp\nu = \sum_{i=1}^n \beta_i \delta_{\theta^\top y_i}$. We recall that the KL divergence is defined as $\mathrm{KL}(p,q) = \int_{-\infty}^{\infty} p(x) \log\left(\frac{p(x)}{q(x)}\right) dx$. Now, we have:

$$
\begin{aligned}
&\mathrm{argmin}_{m_1,\sigma_1^2} \mathrm{KL}(\theta\sharp\mu, \mathcal{N}(m_1,\sigma_1^2)) \\
&= \mathrm{argmax}_{m_1,\sigma_1^2} \left[\sum_{i=1}^n \alpha_i \log\left(\frac{1}{\sqrt{2\pi\sigma_1^2}} \exp\left(-\frac{1}{2\sigma_1^2}(\theta^\top x_i - m_1)^2\right)\right)\right] \\
&= \mathrm{argmax}_{m_1,\sigma_1^2} \left[\sum_{i=1}^n \left(-\frac{\alpha_i}{2\sigma_1^2}(\theta^\top x_i - m_1)^2 - \frac{\alpha_i}{2}\log(\sigma_1^2)\right)\right] \\
&= \mathrm{argmax}_{m_1,\sigma_1^2} f(m_1,\sigma_1^2).
\end{aligned}
$$

Taking the derivatives and setting them to 0, we have:

$$
\begin{aligned}
\frac{d}{dm_1} f(m_1,\sigma_1^2) &= \sum_{i=1}^n -\frac{\alpha_i}{\sigma_1^2}(m_1 - \theta^\top x_i) = -\frac{1}{\sigma_1^2}\left(\sum_{i=1}^n \alpha_i m_1 - \sum_{i=1}^n \alpha_i \theta^\top x_i\right) = 0, \\
\frac{d}{d\sigma_1^2} f(m_1,\sigma_1^2) &= \sum_{i=1}^n \left(\frac{\alpha_i}{2\sigma_1^4}(\theta^\top x_i - m_1)^2 - \frac{\alpha_i}{2\sigma_1^2}\right) = \frac{1}{2\sigma_1^2}\left(\sum_{i=1}^n \frac{\alpha_i}{\sigma_1^2}(\theta^\top x_i - m_1)^2 - \sum_{i=1}^n \alpha_i\right) = 0.
\end{aligned}
$$

Hence, we obtain:

$$
\begin{aligned}
m_1 &= \frac{\sum_{i=1}^n \alpha_i \theta^\top x_i}{\sum_{i=1}^n \alpha_i} = \sum_{i=1}^n \alpha_i \theta^\top x_i, \\
\sigma_1^2 &= \frac{\sum_{i=1}^n \alpha_i (\theta^\top x_i - m_1)^2}{\sum_{i=1}^n \alpha_i} = \sum_{i=1}^n \alpha_i (\theta^\top x_i - m_1)^2.
\end{aligned}
$$

With similar derivation, we obtain the result for m_2, σ_2^2 and complete the proof.

A.2 Proof of Proposition 2

We recall that:

$$
\begin{aligned}
&C_{low}(\theta;\mu,\nu) = (m_1(\theta;\mu) - m_2(\theta;\nu))^2,\\
&C_{up}(\theta;\mu,\nu) = (m_1(\theta;\mu) - m_2(\theta;\nu))^2 + \sigma_1(\theta;\mu)^2 + \sigma_2(\theta;\nu)^2,\\
&\mathrm{W}_2^2(\mathcal{N}(m_1(\theta;\mu),\sigma_1^2(\theta;\mu)),\mathcal{N}(m_2(\theta;\nu),\sigma_2^2(\theta;\nu))) = (m_1(\theta;\mu) - m_2(\theta;\nu))^2 + (\sigma_1(\theta;\mu) - \sigma_2(\theta;\nu))^2.
\end{aligned}
$$

Since $0 \leq (\sigma_1(\theta;\mu) - \sigma_2(\theta;\nu))^2 \leq \sigma_1(\theta;\mu)^2 + \sigma_2(\theta;\nu)^2$, we obtain the result by adding $(m_1(\theta;\mu) - m_2(\theta;\nu))^2$ to the inequalities.

A.3 Proof of Proposition 3

We have μ and ν are two empirical measures i.e., $\mu = \sum_{i=1}^n \alpha_i \delta_{x_i}$ $(\sum_{i=1}^n \alpha_i = 1)$ and $\nu = \sum_{i=1}^m \beta_i \delta_{y_i}$ $(\sum_{i=1}^m \beta_i = 1)$. Now, we have:

$$
\begin{aligned}
&\mathrm{argmin}_{\mathbf{m_1},\sigma_1^2} \mathrm{KL}(\mu, \mathcal{N}(\mathbf{m_1}, \sigma_1^2\mathbf{I}))\\
&= \mathrm{argmax}_{\mathbf{m_1},\sigma_1^2} \sum_{i=1}^n \alpha_i \log\left(\frac{1}{\sqrt{(2\pi)^d|\sigma_1^2\mathbf{I}|}} \exp\left(-\frac{1}{2}(x_i - \mathbf{m_1})^\top |\sigma_1^2\mathbf{I}|^{-1}(x_i - \mathbf{m_1})\right)\right)\\
&= \mathrm{argmax}_{\mathbf{m_1},\sigma_1^2} \sum_{i=1}^n \alpha_i \left(-\frac{1}{2\sigma_1^{2d}}(x_i - \mathbf{m_1})^\top (x_i - \mathbf{m_1}) - \frac{d}{2}\log(\sigma_1^2)\right)\\
&= \mathrm{argmax}_{\mathbf{m_1},\sigma_1^2} f(\mathbf{m}_1, \sigma_1^2)
\end{aligned}
$$

Taking the derivatives and setting them to 0, we have:

$$
\begin{aligned}
\nabla_{\mathbf{m_1}} f(\mathbf{m}_1, \sigma_1^2) &= \sum_{i=1}^n -\frac{\alpha_i}{\sigma_1^{2d}}(\mathbf{m_1} - x_i) = -\frac{1}{\sigma_1^{2d}}\left(\sum_{i=1}^n \alpha_i \mathbf{m_1} - \alpha_i x_i\right) = 0,\\
\frac{d}{d\sigma_1^2} f(\mathbf{m_1}, \sigma_1^2) &= \sum_{i=1}^n \alpha_i \left(\frac{d}{2\sigma_1^{2d+2}}(x_i - \mathbf{m_1})^\top (x_i - \mathbf{m_1}) - \frac{d}{2\sigma_1^2}\right),\\
&= \frac{d}{d\sigma_1^2}\sum_{i=1}^n \left(\frac{\alpha_i \|x_i - \mathbf{m_1}\|_2^2}{\sigma_1^{2d}} - \alpha_i\right) = 0.
\end{aligned}
$$

Hence, we obtain:

$$
\begin{aligned}
\mathbf{m_1} &= \frac{\sum_{i=1}^n \alpha_i x_i}{\sum_{i=1}^n \alpha_i} = \sum_{i=1}^n \alpha_i x_i,\\
\sigma_1^2 &= \left(\frac{\sum_{i=1}^n \alpha_i \|x_i - \mathbf{m}_1\|_2^2}{\sum_{i=1}^n \alpha_i}\right)^{\frac{1}{d}} = \left(\sum_{i=1}^n \alpha_i \|x_i - \mathbf{m}_1\|_2^2\right)^{\frac{1}{d}}.
\end{aligned}
$$

Similarly, we obtain $\mathbf{m_2} = \sum_{i=1}^m \beta_i y_i$ and $\sigma_2^2 = \left(\sum_{i=1}^m \beta_i \|y_i - \mathbf{m}_2\|_2^2\right)^{\frac{1}{d}}$.
Using the linearity of Gaussian distributions, we have $\theta \sharp \mathcal{N}(\mathbf{m_1}, \sigma_1^2\mathbf{I}) = \mathcal{N}(\theta^\top \mathbf{m_1}, \sigma_1^2 \theta^\top \mathbf{I}\theta) =$

Algorithm 1 The conventional estimator of sliced Wasserstein distance.

Input: Probability measures $\mu = \sum_{i=1}^n \alpha_i \delta_{x_i}$ and $\nu = \sum_{i=1}^m \beta_i \delta_{y_i}$, $p \geq 1$, and the number of projections L.
Set $\widehat{\mathrm{SW}}_p^p(\mu, \nu; L) = 0$
for $l = 1$ to L **do**
  Sample $\theta_l \sim \mathcal{U}(\mathbb{S}^{d-1})$
  Compute $\widehat{\mathrm{SW}}_p^p(\mu, \nu; L) = \widehat{\mathrm{SW}}_p^p(\mu, \nu; L) + \frac{1}{L} \int_0^1 |F^{-1}_{\theta_l \sharp \mu}(z) - F^{-1}_{\theta_l \sharp \nu}(z)|^p dz$
end for
Return: $\widehat{\mathrm{SW}}_p^p(\mu, \nu; L)$

$\mathcal{N}(\theta^\top \mathbf{m_1}, \sigma_1^2)$ ($\theta^\top \theta = 1$). Similarly, we obtain $\theta \sharp \mathcal{N}(\mathbf{m_2}, \sigma_2^2 \mathbf{I}) = \mathcal{N}(\theta^\top \mathbf{m_2}, \sigma_2^2)$. Therefore, we have:

$$\mathrm{W}_2^2(\theta \sharp \mathcal{N}(\mathbf{m_1}, \sigma_1^2 \mathbf{I}), \theta \sharp \mathcal{N}(\mathbf{m_2}, \sigma_2^2 \mathbf{I})) = (\theta^\top \mathbf{m_1} - \theta^\top \mathbf{m_2})^2 + (\sigma_1 - \sigma_2)^2.$$

Calculating the expectation, we have:

$$\begin{aligned}
\mathbb{E}\left[\mathrm{W}_2^2(\theta \sharp \mathcal{N}(\mathbf{m_1}, \sigma_1^2 \mathbf{I}), \theta \sharp \mathcal{N}(\mathbf{m_2}, \sigma_2^2 \mathbf{I}))\right] &= (\mathbf{m_1} - \mathbf{m_2})^\top \mathbb{E}[\theta \theta^\top](\mathbf{m_1} - \mathbf{m_2}) + (\sigma_1 - \sigma_2)^2 \\
&= \frac{1}{d} \|\mathbf{m_1} - \mathbf{m_2}\|_2^2 + (\sigma_1 - \sigma_2)^2.
\end{aligned}$$

Let denote $C(\theta) = \mathrm{W}_2^2(\theta \sharp \mathcal{N}(\mathbf{m_1}, \sigma_1^2 \mathbf{I}), \theta \sharp \mathcal{N}(\mathbf{m_2}, \sigma_2^2 \mathbf{I}))$, we have:

$$\begin{aligned}
C(\theta) - \mathbb{E}[C(\theta)] &= (\theta^\top \mathbf{m_1} - \theta^\top \mathbf{m_2})^2 - \frac{1}{d} \|\mathbf{m_1} - \mathbf{m_2}\|_2^2 \\
&= \left(\theta^\top \sum_{i=1}^n \alpha_i x_i - \theta^\top \sum_{i=1}^m \beta_i y_i\right)^2 - \frac{1}{d} \left\| \sum_{i=1}^n \alpha_i x_i - \sum_{i=1}^m \beta_i y_i \right\|_2^2 \\
&= \left(\sum_{i=1}^n \alpha_i \theta^\top x_i - \sum_{i=1}^m \beta_i \theta^\top y_i\right)^2 - \frac{1}{d} \left\| \sum_{i=1}^n \alpha_i x_i - \sum_{i=1}^m \beta_i y_i \right\|_2^2 \\
&= C_{low}(\theta; \mu, \nu) - \mathbb{E}[C_{low}(\theta; \mu, \nu)].
\end{aligned}$$

Similarly, we obtain:

$$\begin{aligned}
&\mathrm{Var}\left[C(\theta)\right] = \mathbb{E}\left[(C(\theta) - \mathbb{E}[C(\theta)])^2\right] = \mathbb{E}\left[(C_{low}(\theta; \mu, \nu) - \mathbb{E}[C_{low}(\theta; \mu, \nu)])^2\right] = \mathrm{Var}[C_{low}(\theta; \mu, \nu)], \\
&\mathrm{Cov}\left[C(\theta), W(\theta; \mu, \nu)\right] = \mathbb{E}\left[(C(\theta) - \mathbb{E}[C(\theta)])(W(\theta; \mu, \nu) - \mathbb{E}[W(\theta; \mu, \nu)])\right] \\
&\qquad = \mathbb{E}\left[(C_{low}(\theta; \mu, \nu) - \mathbb{E}[C_{low}(\theta; \mu, \nu)])(W(\theta; \mu, \nu) - \mathbb{E}[W(\theta; \mu, \nu)])\right] \\
&\qquad = \mathrm{Cov}\left[C_{low}(\theta; \mu, \nu), W(\theta; \mu, \nu)\right].
\end{aligned}$$

Therefore, it is sufficient to claim that using the C_{low} control variate is equivalent to use $\mathrm{W}_2^2(\theta \sharp \mathcal{N}(\mathbf{m_1}, \sigma_1^2 \mathbf{I}), \theta \sharp \mathcal{N}(\mathbf{m_2}, \sigma_2^2 \mathbf{I}))$ as the control variate.

B Algorithms

We present the algorithm for computing the conventional Monte Carlo estimator of the sliced Wasserstein distance between two discrete measures in Algorithm 1. Similarly, we provide the

Algorithm 2 The lower bound control variate estimator of sliced Wasserstein distance.

Input: Probability measures $\mu = \sum_{i=1}^n \alpha_i \delta_{x_i}$ and $\nu = \sum_{i=1}^m \beta_i \delta_{y_i}$, $p \geq 1$, and the number of projections L.
Set $\widehat{\text{SW}}_p^p(\mu, \nu; L) = 0$
Compute $\bar{x} = \sum_{i=1}^n \alpha_i x_i$, and $\bar{y} = \sum_{i=1}^m \beta_i y_i$
for $l = 1$ to L **do**
  Sample $\theta_l \sim \mathcal{U}(\mathbb{S}^{d-1})$
  Compute $w_l = \int_0^1 |F^{-1}_{\theta_l \sharp \mu}(z) - F^{-1}_{\theta_l \sharp \nu}(z)|^p dz$
  Compute $c_l = (\theta^\top \bar{x} - \theta^\top \bar{y})^2$
end for
Compute $\widehat{\text{SW}}_p^p(\mu, \nu; L) = \frac{1}{L} \sum_{l=1}^L w_l$
Compute $b = \frac{1}{d} \|\bar{x} - \bar{y}\|_2^2$
Compute $\gamma = \frac{\frac{1}{L} \sum_{l=1}^L (w_l - \widehat{\text{SW}}_p^p(\mu, \nu; L))(c_l - b)}{\frac{1}{L} \sum_{l=1}^L (c_l - b)^2}$
Compute $\widehat{\text{LCV-SW}}_p^p(\mu, \nu; L) = \widehat{\text{SW}}_p^p(\mu, \nu; L) - \gamma \frac{1}{L} \sum_{l=1}^L (c_l - b)$
Return: $\widehat{\text{LCV-SW}}_p^p(\mu, \nu; L)$

Algorithm 3 The upper bound control variate estimator of sliced Wasserstein distance.

Input: Probability measures $\mu = \sum_{i=1}^n \alpha_i \delta_{x_i}$ and $\nu = \sum_{i=1}^m \beta_i \delta_{y_i}$, $p \geq 1$, and the number of projections L.
Set $\widehat{\text{SW}}_p^p(\mu, \nu; L) = 0$
Compute $\bar{x} = \sum_{i=1}^n \alpha_i x_i$, and $\bar{y} = \sum_{i=1}^m \beta_i y_i$
for $l = 1$ to L **do**
  Sample $\theta_l \sim \mathcal{U}(\mathbb{S}^{d-1})$
  Compute $w_l = \int_0^1 |F^{-1}_{\theta_l \sharp \mu}(z) - F^{-1}_{\theta_l \sharp \nu}(z)|^p dz$
  Compute $c_l = (\theta^\top \bar{x} - \theta^\top \bar{y})^2 + \sum_{i=1}^n \alpha_i (\theta^\top x_i - \theta^\top \bar{x})^2 + \sum_{i=1}^m \beta_i (\theta^\top y_i - \theta^\top \bar{y})^2$
end for
Compute $\widehat{\text{SW}}_p^p(\mu, \nu; L) = \frac{1}{L} \sum_{l=1}^L w_l$
Compute $b = \frac{1}{d} \|\bar{x} - \bar{y}\|_2^2 + \frac{1}{d} \sum_{i=1}^n \alpha_i \|(x_i - \bar{x})\|_2^2 + \frac{1}{d} \sum_{i=1}^m \beta_i \|(y_i - \bar{y})\|_2^2$
Compute $\gamma = \frac{\frac{1}{L} \sum_{l=1}^L (w_l - \widehat{\text{SW}}_p^p(\mu, \nu; L))(c_l - b)}{\frac{1}{L} \sum_{l=1}^L (c_l - b)^2}$
Compute $\widehat{\text{UCV-SW}}_p^p(\mu, \nu; L) = \widehat{\text{SW}}_p^p(\mu, \nu; L) - \gamma \frac{1}{L} \sum_{l=1}^L (c_l - b)$
Return: $\widehat{\text{UCV-SW}}_p^p(\mu, \nu; L)$

algorithms for the lower bound control variate estimator and the upper bound control variate estimator in Algorithm 2 and in Algorithm 3 respectively.

C Related Works

Sliced Wasserstein variants with different projecting functions. The conventional sliced Wasserstein is based on a linear projecting function i.e., the inner product to project measures to one dimension. In some special cases of probability measures, some other projecting functions

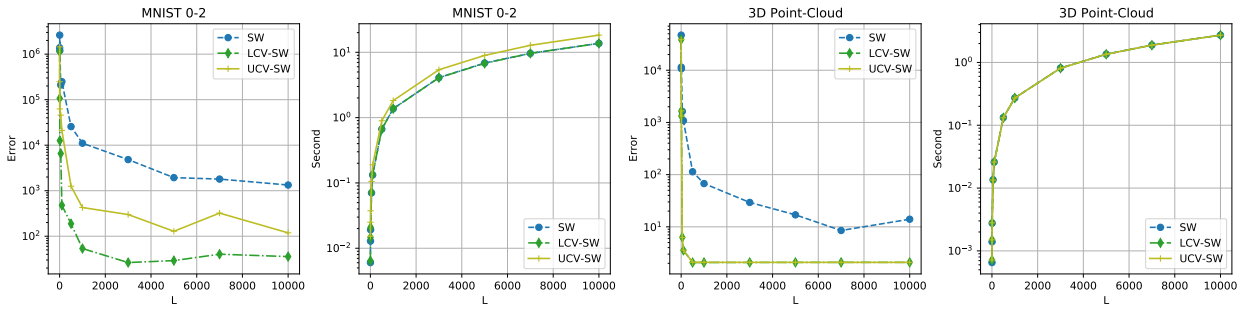

Figure 4: The empirical errors of the conventional estimator (SW) and the control variate estimators (LCV-SW, UCV-SW) when comparing empirical distributions over MNIST images and point-clouds.

Table 3: Estimated variances of the conventional estimator's and the control variate estimators.

Estimator	MNIST 0-1	MNIST 0-2	Point-cloud-1	Point-cloud-2
SW	4711.97 ± 70.99	1398 ± 18.05	10.72 ± 0.47	10.67 ± 0.34
LCV-SW	$\mathbf{48.93 \pm 1.02}$	$\mathbf{4.00 \pm 0.16}$	$\mathbf{0.616 \pm 0.001}$	$\mathbf{0.11 \pm 0.0001}$
UCV-SW	243.98 ± 2.47	268.67 ± 2.21	1.07 ± 0.47	1.07 ± 0.34

might be preferred e.g., generalized sliced Wasserstein [12] distance with circular, polynomial projecting function, hierarchical sliced Wasserstein [28] with the composition of linear projecting function, spherical sliced Wasserstein [2] with geodesic spherical projecting function, convolution sliced Wasserstein with convolution projecting function [24], and so on. Despite having different projecting functions, all mentioned sliced Wasserstein variants utilize random projecting directions that follow the uniform distribution over the unit hypersphere and are estimated using the Monte Carlo integration scheme. Therefore, we could directly adapt the proposed control variates in these variants.

Applications of the control variate estimators. Since the proposed control variates are used to estimate the sliced Wasserstein distance, they can be applied to all applications where sliced Wasserstein exists. We would like to mention some other applications such as domain adaptation [16], approximate Bayesian computation [20], color transfer [17], and so on.

D Additional Experiments

In this section, we provide some additional experiments for applications in the main text. In particular, we calculate the empirical variances of the conventional estimator and the control variate estimators on different images of digits and point-clouds in Appendix D.1. We provide the point-cloud gradient flow between two new point-cloud in Appendix D.2. We then provide the detailed training of deep generative models and additional generated images in Appendix D.3.

D.1 Comparing empirical probability measures over images and point-clouds

Settings. We follow the same settings which are used in the main text. However, for MNIST, we compare probability measures over digit 0 and digit 2. For point-clouds, we compare two different point-

Table 4: Summary of Wasserstein-2 scores (multiplied by 10^4) from 3 different runs, computational time in second (s) to reach step 500 of different sliced Wasserstein variants in gradient flows.

Distances	Step 3000 ($W_2 \downarrow$)	Step 4000 ($W_2 \downarrow$)	Step 5000 ($W_2 \downarrow$)	Step 6000($W_2 \downarrow$)	Step 8000 ($W_2 \downarrow$)	Time (s $\downarrow$)
SW L=10	305.3907 ± 0.4919	137.7762 ± 0.3630	36.1807 ± 0.1383	0.1054 ± 0.0022	$2.3e-5 \pm 1.0e-5$	$\mathbf{26.30 \pm 0.03}$
LCV-SW L=10	302.9718 ± 0.1788	$\mathbf{135.9132 \pm 0.0922}$	$\mathbf{35.0292 \pm 0.1457}$	0.0452 ± 0.0045	$\mathbf{1.7e-5 \pm 0.3e-5}$	27.65 ± 0.01
UCV-SW L=10	$\mathbf{302.9717 \pm 0.1788}$	$\mathbf{135.9132 \pm 0.0922}$	35.0295 ± 0.1458	$\mathbf{0.0446 \pm 0.0038}$	$2.0e-5 \pm 0.4e-5$	29.56 ± 0.01
SW L=100	300.6303 ± 0.2375	134.0492 ± 0.3146	33.8608 ± 0.1348	0.0121 ± 0.0010	$1.6e-5 \pm 0.2e-5$	$\mathbf{222.06 \pm 1.34}$
LCV-SW L=100	300.2362 ± 0.0054	133.5238 ± 0.0065	33.4460 ± 0.0030	$0.0084 \pm 5.8e-5$	$\mathbf{1.4e-5 \pm 0.1e-5}$	223.79 ± 0.82
UCV-SW L=100	$\mathbf{300.2631 \pm 0.0054}$	$\mathbf{133.5237 \pm 0.0065}$	$\mathbf{33.4459 \pm 0.0030}$	$\mathbf{0.0083 \pm 8.5e-5}$	$1.6e-5 \pm 0.1e-5$	235.29 ± 1.65

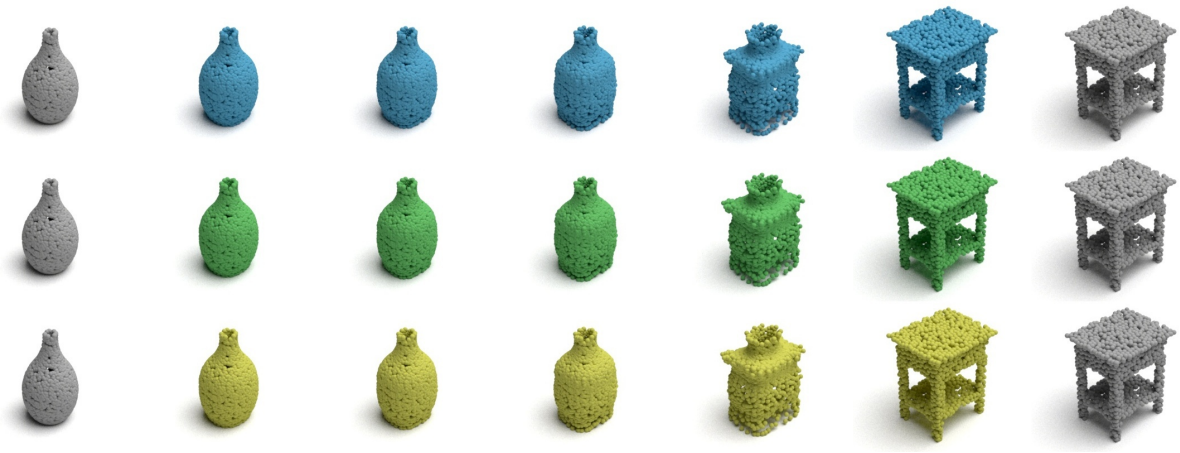

Figure 5: Point-Cloud gradient flows from SW, LCV-SW, and UCV-SW.

clouds from the main text. We report the estimated variances of $W(\theta; \mu, \nu)$, $Z_{low}(\theta; \mu, \nu)$, $Z_{up}(\theta; \mu, \nu)$ defined in Equation 6, Equation 22, and Equation 23 in turn in Table 3. We use a large number of samples e.g., 50000 Monte Carlo samples. In the table, point-cloud-1 denotes the pair in Figure 2 and point-cloud-2 denotes the pair in Figure 5.

Results. We show the estimated errors and the corresponding computational time in Figure 4. From the figure, we observe the same phenomenon as in the main text. In particular, the control variate estimators reduce considerably the errors in both cases while having approximately the same computational time.

D.2 Point-cloud Gradient Flows

Settings. We follow the same settings which are used in the main text. However, we use different point-clouds which are also used in Appendix D.1.

Results. We show the quantitative results in Table 4 and the corresponding qualitative result in Figure 5. Overall, we observe the same phenomenon as in the main text. In particular, the control variate estimators i.e., LCV-SW, UCV-SW help to drive the flows to converge faster to the target point-cloud than the conventional estimator i.e., SW. It is worth noting that the computational time of the control variate estimators is only slightly higher than the conventional estimator for both settings of the number of projections $L = 10, 100$.

SW L=1000 LCV-SW L=1000 UCV-SW L=1000

Figure 6: Random generated images of distances on CIFAR10 and CelebA.

D.3 Deep Generative Modeling

Setting. We denote μ as our data distribution. We then design the model distribution ν_ϕ as a push forward probability measure that is created by pushing a unit multivariate Gaussian (ϵ) through a neural network G_ϕ that maps from the realization of the noise to the data space. We use a second neural network T_β that maps from data space to a single scalar. We denote T_{β_1} is the sub neural network of T_β that maps from the data space to a feature space (output of the last Resnet block), and T_{β_2} that maps from the feature space (image of T_{β_1}) to a single scalar. More precisely, $T_\beta = T_{\beta_2} \circ T_{\beta_1}$. We use the following neural networks for G_ϕ and T_β:

- **CIFAR10**:
  - G_ϕ: $z \in \mathbb{R}^{128} (\sim \epsilon : \mathcal{N}(0,1)) \rightarrow 4 \times 4 \times 256$(Dense, Linear) $\rightarrow$ ResBlock up 256 $\rightarrow$ ResBlock up 256 $\rightarrow$ ResBlock up 256 $\rightarrow$ BN, ReLU, $\rightarrow 3 \times 3$ conv, 3 Tanh .
  - T_{β_1}: $x \in [-1,1]^{32\times32\times3} \rightarrow$ ResBlock down 128 $\rightarrow$ ResBlock down 128 $\rightarrow$ ResBlock down 128 $\rightarrow$ ResBlock 128 $\rightarrow$ ResBlock 128.
  - T_{β_2}: $\boldsymbol{x} \in \mathbb{R}^{128\times8\times8} \rightarrow$ ReLU $\rightarrow$ Global sum pooling(128) $\rightarrow$ 1(Spectral normalization).
  - $T_\beta(x) = T_{\beta_2}(T_{\beta_1}(x))$.
- **CelebA:**

- G_ϕ: $z \in \mathbb{R}^{128} (\sim \epsilon : \mathcal{N}(0,1)) \rightarrow 4 \times 4 \times 256$(Dense, Linear) $\rightarrow$ ResBlock up 256 $\rightarrow$ ResBlock up 256 $\rightarrow$ ResBlock up 256 $\rightarrow$ ResBlock up 256 $\rightarrow$ BN, ReLU, $\rightarrow 3 \times 3$ conv, 3 Tanh .
- T_{β_1}: $\boldsymbol{x} \in [-1,1]^{32\times32\times3} \rightarrow$ ResBlock down 128 $\rightarrow$ ResBlock down 128 $\rightarrow$ ResBlock down 128 $\rightarrow$ ResBlock 128 $\rightarrow$ ResBlock 128.
- T_{β_2}: $\boldsymbol{x} \in \mathbb{R}^{128\times8\times8} \rightarrow$ ReLU $\rightarrow$ Global sum pooling(128) $\rightarrow$ 1(Spectral normalization).
- $T_\beta(x) = T_{\beta_2}(T_{\beta_1}(x))$.

We use the following bi-optimization problem to train our neural networks:

$$\min_{\beta_1,\beta_2} \left(\mathbb{E}_{x\sim\mu}[\min(0,-1+T_\beta(x))] + \mathbb{E}_{z\sim\epsilon}[\min(0,-1-T_\beta(G_\phi(z)))]\right),$$
$$\min_{\phi} \mathbb{E}_{X\sim\mu^{\otimes m},Z\sim\epsilon^{\otimes m}}[\mathcal{S}(\tilde{T}_{\beta_1,\beta_2}\sharp P_X, \tilde{T}_{\beta_1,\beta_2}\sharp G_\phi\sharp P_Z)],$$

where the function $\tilde{T}_{\beta_1,\beta_2} = [T_{\beta_1}(x), T_{\beta_2}(T_{\beta_1}(x))]$ which is the concatenation vector of $T_{\beta_1}(x)$ and $T_{\beta_2}(T_{\beta_1}(x))$, $\mathcal{S}$ is an estimator of the sliced Wasserstein distance. The number of training iterations is set to 100000 on CIFAR10 and 50000 in CelebA. We update the generator G_ϕ every 5 iterations and we update the feature function T_β every iteration. The mini-batch size m is set to 128 in all datasets. We use the Adam [11] optimizer with parameters $(\beta_1, \beta_2) = (0, 0.9)$ for both G_ϕ and T_β with the learning rate 0.0002. We use 50000 random samples from estimated generative models G_ϕ for computing the FID scores and the Inception scores. In evaluating FID scores, we use all training samples for computing statistics of datasets.

Results. In addition to the result in the main text, we provide generated images from the conventional estimator (SW) and the control variate estimator (LCV-SW and UCV-SW) with the number of projections $L = 1000$ in Figure 6. Overall, we see that by increasing the number of projections, the generated images are visually improved for all estimators. This result is consistent with the FID scores and the IS scores in Table 2.

E Computational Infrastructure

For comparing empirical probability measures over images and point-cloud application, and the point-cloud gradient flows application, we use a Macbook Pro M1 for conducting experiments. For deep generative modeling, experiments are run on a single NVIDIA V100 GPU.