FAST ESTIMATION OF WASSERSTEIN DISTANCES VIA REGRESSION ON SLICED WASSERSTEIN DISTANCES

Anonymous authors
Paper under double-blind review

ABSTRACT

We address the problem of efficiently computing Wasserstein distances for multiple pairs of distributions drawn from a meta-distribution. To this end, we propose a fast estimation method based on regressing Wasserstein distance on sliced Wasserstein (SW) distances. Specifically, we leverage both standard SW distances, which provide lower bounds, and lifted SW distances, which provide upper bounds, as predictors of the true Wasserstein distance. To ensure parsimony, we introduce two linear models: an unconstrained model with a closed-form least-squares solution, and a constrained model that uses only half as many parameters. We show that accurate models can be learned from a small number of distribution pairs. Once estimated, the model can predict the Wasserstein distance for any pair of distributions via a linear combination of SW distances, making it highly efficient. Empirically, we validate our approach on diverse tasks, including Gaussian mixtures, point-cloud classification, and Wasserstein-space visualizations for 3D point clouds. Across various datasets such as MNIST point clouds, ShapeNetV2, MERFISH Cell Niches, and scRNA-seq, our method consistently provides a better approximation of Wasserstein distance than the state-of-the-art Wasserstein embedding model, Wasserstein Wormhole, particularly in low-data regimes. Finally, we demonstrate that our estimator can also accelerate Wormhole training, yielding *RG-Wormhole*.

1 INTRODUCTION

Optimal Transport (OT) and Wasserstein distances (Villani, 2009; Peyré & Cuturi, 2019) have become essential tools in machine learning, widely used for quantifying the similarity or dissimilarity between probability distributions. Fundamentally, the Wasserstein distance measures the minimum cost required to "transport" mass from one distribution to another, effectively capturing the underlying geometry of the data. Thanks to their clear geometric interpretation and mathematical robustness, Wasserstein distances have found applications across various fields, such as generative modeling Genevay et al. (2018), computational biology Bunne et al. (2023), chemistry Wu et al. (2023), and image processing Feydy et al. (2017). Despite its utility, computing the exact Wasserstein distance is computationally expensive. It typically requires solving a large-scale linear program to find an optimal transport plan, with a time complexity of $\mathcal{O}(n^3 \log n)$ for discrete distributions of size n. This high cost severely limits its use in large-scale or real-time settings.

In many applications, Wasserstein distances are computed (repeatedly) for many pairs of distributions, e.g., dataset comparisons (Alvarez-Melis & Fusi, 2020), 3D point-cloud autoencoder (Achlioptas et al., 2018), point-cloud nearest neighbor classification/regression (Rubner et al., 1998), learning embeddings for distributions (Kolouri et al., 2021), density-density regression (Chen et al., 2023), and so on. Therefore, the high computational complexities of the Wasserstein distance become the main bottleneck to scaling up these applications. As a result, speeding up the computation of the Wasserstein distance has become a vital task in practice.

To address this bottleneck, a straightforward improvement is to speed up the computation of the Wasserstein distance. For example, entropic regularization (Cuturi, 2013) enables fast approximation via Sinkhorn iterations, while other methods exploit the structure in the transport plan, such as low-rank approximations (Scetbon et al., 2021). In addition, some approaches rely on strong structural assumptions, such as the Bures-Wasserstein metric (Dowson & Landau, 1982) gives a closed-form solution for the exact 2-Wasserstein distance (W_2) under the Gaussian assumption on distributions.

Another approach is to cast computing Wasserstein distances for many pairs of distributions as a learning problem, i.e., learning a model first to predict the Wasserstein distance given any pair of distributions, then use the model later for the mentioned downstream tasks. For example, Deep Wasserstein Embedding (DWE) (Courty et al., 2018) trains a Siamese convolutional network to match OT distances between 2D images, while Wasserstein Wormhole (Haviv et al., 2024) employs transformer-based architectures to learn embeddings of distributions, allowing Euclidean distances in the learned space to approximate Wasserstein distances efficiently. While effective, these deep learning-based methods require significant computational resources and time to train, and their performance may degrade when limited training data are available. Moreover, these approaches are limited to empirical distributions because of the use of neural networks.

In this work, we propose a novel approach to predict the Wasserstein distance without relying on any neural networks or learned embeddings. Moreover, the proposed approach relies on a parsimonious model and can handle both continuous and discrete distributions. In particular, we propose to regress the Wasserstein distance on sliced Wasserstein (SW) distances (Rabin et al., 2010; Mahey et al., 2023; Nguyen & Ho, 2023; Liu et al., 2025; Deshpande et al., 2019; Rowland et al., 2019). In greater detail, we introduce linear models with Wasserstein distances as the response and SW distances as the predictors. We provide estimates of the models via efficient least-squares estimates. In addition, since sliced Wasserstein distances have low computational complexity, the resulting Wasserstein regressor is computationally efficient.

Contribution: In summary, our main contributions are three-fold:

1. We introduce the first regression framework where the Wasserstein distance serves as the response variable and various sliced Wasserstein (SW) distances act as predictors, in the setting of random pairs of distributions. This framework not only uncovers the relationship between the Wasserstein distance and its SW-based approximations but also enables efficient estimation of the Wasserstein distance. Specifically, we use SW distance (Bonneel et al., 2015), Max-SW (Deshpande et al., 2019), and energy-based SW (Nguyen & Ho, 2023), all of which provide lower bounds on the Wasserstein distance, as predictors. In addition, we incorporate lifted SW distances, which provide upper bounds, including projected Wasserstein (Rowland et al., 2019), Minimum SW generalized geodesics (Mahey et al., 2023), and expected sliced distance (Liu et al., 2025).

2. We propose two linear models for the regression problem and describe their estimation via least-squares. The first model is unconstrained and admits a closed-form least-squares solution. The second model incorporates constraints that leverage the known bounds between SW distances and the Wasserstein distance, thereby reducing the number of parameters by half. Based on these estimations, we obtain a fast method to approximate the Wasserstein distance for any pair of distributions, with the same computational complexity as that of computing SW distances.

3. Empirically, we demonstrate that our approach yields accurate estimates of the Wasserstein distance, particularly in low-data regimes. We first evaluate its accuracy through simulations with Gaussian mixtures. We then apply the estimated distances to visualize distributional data and to perform k-NN classification on ShapeNetV2 point clouds. Next, we benchmark our method against Wasserstein Wormhole, the state-of-the-art Wasserstein embedding model, across four datasets of increasing dimensionality: MNIST point clouds, ShapeNetV2, MERFISH cell niches, and scRNA-seq. Finally, we propose *RG-Wormhole*, a variant of Wasserstein Wormhole that replaces its Wasserstein computations with our estimates, preserving accuracy while substantially reducing training time.

Organization. Section 2 reviews preliminaries on the Wasserstein distance, its sliced variants, and their computation. Section 3 introduces our regression framework for approximating Wasserstein distances from sliced variants, together with both constrained and unconstrained linear models. Section 4 reports the experimental results. The appendices provide supplementary experiments (mixtures of Gaussians and distributional space visualizations), detailed experimental settings, theoretical proofs, and additional related work.

Notations. For any $d \geq 2$, let $\mathbb{S}^{d-1} := \{\theta \in \mathbb{R}^d : \|\theta\|_2 = 1\}$ denote the unit sphere in $\mathbb{R}^d$, and let $\mathcal{U}(\mathbb{S}^{d-1})$ denote the uniform distribution on it. For $p \geq 1$, we write $\mathcal{P}_p(\mathcal{X})$ for the set of all probability measures on $\mathcal{X}$ with the finite p th moment. Given two sequences a_n and b_n, the notation $a_n = \mathcal{O}(b_n)$ means that $a_n \leq Cb_n$ for all $n \geq 1$, for some universal constant $C > 0$. For a measurable map P, the notation $P\sharp\mu$ denotes the push-forward of μ through P. Additional notation will be introduced as needed.

2 PRELIMINARIES

We first review definitions and computational aspects of the Wasserstein distance and its related properties in one dimension.

Wasserstein distance. Wasserstein-p ($p \geq 1$) distance Villani (2008); Peyré et al. (2019) between two distributions $\mu \in \mathcal{P}_p(\mathbb{R}^d)$ and $\nu \in \mathcal{P}_p(\mathbb{R}^d)$ (dimension $d \geq 1$) is defined as:

$$W_p^p(\mu, \nu) = \inf_{\pi \in \Pi(\mu,\nu)} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|x - y\|_p^p d\pi(x, y), \tag{1}$$

where $\Pi(\mu, \nu) = \left\{\pi \in \mathcal{P}(\mathbb{R}^d \times \mathbb{R}^d)\right\} \mid \int_{\mathbb{R}^d} d\pi(x, y) = \mu(x), \int_{\mathbb{R}^d} d\pi(x, y) = \nu(y)\Big\}$ is the set of all transportation plans i.e., joint distributions which have marginals be two comparing distributions. When μ and ν are discrete distributions i.e., $\mu = \sum_{i=1}^n \alpha_i \delta_{x_i}$ ($n \geq 1$) and $\nu = \sum_{j=1}^m \beta_j \delta_{y_j}$ ($m \geq 1$) where $\sum_{i=1}^N \alpha_i = \sum_{j=1}^m \beta_j = 1$ and $\alpha_i \geq 0, \beta_j \geq 0$ for all $i = 1, \ldots, n$ and $j = 1, \ldots, m$, Wasserstein distance between μ and ν defined as: $W_p^p(\mu, \nu) = \min_{\gamma \in \Gamma(\alpha,\beta)} \sum_{i=1}^n \sum_{j=1}^m \|x_i - y_j\|_p^p \gamma_{ij}$, where $\Gamma(\alpha, \beta) = \{\gamma \in \mathbb{R}_+^{n \times m} \mid \gamma \mathbf{1} = \alpha, \gamma^\top \mathbf{1} = \beta\}$. Without loss of generality, we assume that $n \geq m$. Therefore, the time complexity for solving this linear programming is $\mathcal{O}(n^3 \log n)$ Peyré & Cuturi (2019) and $\mathcal{O}(n^2)$, which are expensive.

One-dimensional Case. When $d = 1$, the Wasserstein distance can be efficiently calculated. For the continuous case, Wasserstein-2 distance has the following form: $W_p^p(\mu, \nu) = \int_0^1 |F_\mu^{-1}(t) - F_\nu^{-1}(t)|^p dt$, where F_μ^{-1} and F_ν^{-1} denote the quantile functions of μ and ν respectively. Here, the transportation plan is $\pi_{(\mu,\nu)} = (F_\mu^{-1}, F_\nu^{-1})\sharp\mathcal{U}([0, 1])$. When μ and ν are discrete distributions, i.e. $\mu = \sum_{i=1}^n \alpha_i \delta_{x_i}$ ($n \geq 1$) and $\nu = \sum_{j=1}^m \beta_j \delta_{y_j}$, quantile functions of μ and ν are:

$$F_\mu^{-1}(t) = \sum_{i=1}^n x_{(i)} I\left(\sum_{j=1}^{i-1} \alpha_{(j)} < t \leq \sum_{j=1}^{i} \alpha_{(j)}\right), F_\nu^{-1}(t) = \sum_{j=1}^m y_{(j)} I\left(\sum_{i=1}^{j-1} \beta_{(i)} < t \leq \sum_{i=1}^{j} \beta_{(i)}\right),$$

where $x_{(1)} \leq \ldots \leq x_{(n)}$ and $y_{(1)} \leq \ldots \leq y_{(m)}$ are the sorted supports (or order statistics). Therefore, the one-dimensional Wasserstein distance can be computed in $\mathcal{O}(n \log n)$ in time and $\mathcal{O}(n)$ in space (assuming that $n > m$).

Random Projection. A key technique that plays a vital role in later discussion is random projection. We consider a function $P_\theta : \mathbb{R}^d \to \mathbb{R}$ where $\theta \sim \sigma(\theta)$ ($\sigma(\theta) \sim \mathcal{P}(\mathbb{S}^{d-1})$) is a random variable. For simplicity, we consider the traditional setup where $\theta \sim \mathcal{U}(\mathbb{S}^{d-1})$ and $P_\theta(x) = \langle \theta, x \rangle$ (Bonneel et al., 2015; Rabin et al., 2012). However, the following discussion holds for any other types of projections (Kolouri et al., 2019; Bonet et al., 2023b; 2025; 2023c). For $\mu \in \mathcal{P}_p(\mathbb{R}^d)$ and $\nu \in \mathcal{P}_p(\mathbb{R}^d)$, one-dimensional projected Wasserstein distance with P_θ is defined as:

$$\underline{W}_p^p(\mu, \nu; P_\theta) = W_p^p(P_\theta \sharp \mu, P_\theta \sharp \nu) = \int_0^1 |F_{P_\theta \sharp \mu}^{-1}(t) - F_{P_\theta \sharp \nu}^{-1}(t)|^p dt. \tag{2}$$

The second approach to construct a Wasserstein-type discrepancy from one-dimensional projection is using lifted transportation plan. There are many ways to construct such lifted plan using disintegration of measures (Muzellec & Cuturi, 2019; Tanguy et al., 2025). In practice, the most used way (Liu et al., 2025; Tanguy et al., 2025) is:

$$\overline{W}_p^p(\mu, \nu; P_\theta) = \int_{\mathbb{R}^d \times \mathbb{R}^d} \|x - y\|_p^p d\pi^\theta(x, y) \tag{3}$$

$$= \int_{\mathbb{R} \times \mathbb{R}} \int_{P_\theta^{-1}(t_1) \times P_\theta^{-1}(t_2)} \|x - y|_p^p d\mu_{t_1} \otimes \nu_{t_2}(x, y) d\pi_{(P_\theta \sharp \mu, P_\theta \sharp \nu)}(t_1, t_2), \tag{4}$$

where $\pi^\theta \in \Pi(\mu, \nu)$ is the lifted transportation plan, $\pi_{(P_\theta \sharp \mu, P_\theta \sharp \nu)}$ is the optimal transport plan between $P_\theta \sharp \mu$ and $P_\theta \sharp \nu$, μ_{t_1} and ν_{t_2} are disintegration of μ and ν at t_1 and t_2 the function P_θ, and $\otimes$ denotes the product of measures. When dealing with discrete measures μ and ν, $\overline{W}_p^p(\mu, \nu; P_\theta)$ can still be computed efficiently (Mahey et al., 2023; Liu et al., 2025) i.e., $\mathcal{O}(n \log n)$ in time and $\mathcal{O}(n)$ in space (assumed that $n > m$). The quantity $\overline{W}_p^p(\mu, \nu; P_\theta)$ is known as lifted cost (Tanguy et al., 2025) or sliced Wasserstein generalized geodesic (Mahey et al., 2023; Liu et al., 2025). From previous work (Nguyen & Ho, 2023; Mahey et al., 2023; Tanguy, 2023), we know the following relationship $\underline{W}_p(\mu, \nu; P_\theta) \leq W_p(\mu, \nu) \leq \overline{W}_p(\mu, \nu; P_\theta)$.

3 REGRESSION OF WASSERSTEIN DISTANCE ONTO SLICED OPTIMAL TRANSPORT DISTANCES

In this section, we present a framework for regressing the Wasserstein distance onto sliced Wasserstein distances, propose some models, and discuss related computational properties.

3.1 SLICED WASSERSTEIN AND LIFTED SLICED WASSERSTEIN

Sliced Wasserstein distances. Given $\mu \in \mathcal{P}_p(\mathbb{R}^d)$ and $\nu \in \mathcal{P}_p(\mathbb{R}^d)$, a sliced Wasserstein-$p$ distance can be defined as follows (Rabin et al., 2012; Nguyen, 2025):

$$SW_p^p(\mu, \nu; \sigma) = \mathbb{E}_{\theta\sim\sigma}\left[\underline{W}_p^p(\mu, \nu; P_\theta)\right], \tag{5}$$

where $P_\theta : \mathbb{R}^d \to \mathbb{R}$ is the projection function, $\underline{W}_p^p(\mu, \nu; P_\theta)$ is the one-dimensional projected Wasserstein distance (equation 2), and $\sigma \in \mathcal{P}(\mathbb{S}^{d-1})$ is the slicing distribution. By changing the slicing distribution, we can obtain variants of SW. There are three main ways: 1. *Fixed prior:* The simplest way is to choose σ to be a fixed and known distribution, e.g., the uniform distribution $\mathcal{U}(\mathbb{S}^{d-1})$ as in the conventional SW (Rabin et al., 2012). 2. *Optimization-based:* We can also find σ that prioritizes some realizations of θ that satisfies a notion of informativeness. For example, σ can put more masses to realizations of θ where $\underline{W}_p^p(\mu, \nu; P_\theta)$ have high value, i.e., setting informativeness as discriminativeness. For example, we can find σ by solving (Nguyen et al., 2021): $\sup_{\sigma\in\mathcal{M}(\mathbb{S}^{d-1})} \mathbb{E}_{\theta\sim\sigma}[\underline{W}_p^p(\mu, \nu; P_\theta)]$, where $\mathcal{M}(\mathbb{S}^{d-1}) \subset \mathcal{P}(\mathbb{S}^{d-1})$ be a set of probability measures on $\mathbb{S}^{d-1}$. When $\mathcal{M}(\mathbb{S}^{d-1}) = \{\delta_\theta \mid \theta \in \mathbb{S}^{d-1}\}$, max sliced Wasserstein distance (Deshpande et al., 2019) is obtained: Max-$SW(\mu, \nu) = \max_{\theta\in\mathbb{S}^{d-1}} \underline{W}_p(\mu, \nu; P_\theta)]$. 3. *Energy-based:* An optimization-free way to select σ is to design it as an energy-based distribution with the unnormalized density: $p_\sigma(\theta) \propto f(\underline{W}_p^p(\mu, \nu; P_\theta))$, where f is often chosen to be an increasing function on the positive real line, i.e., an exponential function. This choice of slicing distribution leads to energy-based SW (EBSW) (Nguyen & Ho, 2023).

Empirical estimation. For SW, Monte Carlo estimation is used to approximate the distance: $\widehat{SW}_p^p(\mu, \nu; \theta_1, \ldots, \theta_L) = \frac{1}{L}\sum_{l=1}^L \underline{W}_p^p(\mu, \nu; P_{\theta_l})$, where $\theta_1, \ldots, \theta_L \overset{i.i.d}{\sim} \mathcal{U}(\mathbb{S}^{d-1})$ ($L > 0$) are projecting directions (other sampling techniques can also be used (Nguyen et al., 2024; Nguyen & Ho, 2024; Sisouk et al., 2025)). For Max-SW, we can use $\hat{\theta}_T$ which is the solution of an optimization algorithm with $T > 0$ iterations, e.g., projected gradient ascent (Nietert et al., 2022) or Riemannian gradient ascent Lin et al. (2020): $\widehat{\text{Max-}SW}_p^p(\mu, \nu; \hat{\theta}_T) = \underline{W}_p^p(\mu, \nu; P_{\hat{\theta}_T})$. For EBSW, one simple way to estimate the distance is to use importance sampling: $\widehat{EBSW}_p^p(\mu, \nu; \theta_1, \ldots, \theta_L) = \sum_{l=1}^L \hat{w}_l \underline{W}_p^p(\mu, \nu; P_{\theta_l})$, where $\hat{w}_l = \frac{f(\underline{W}_p^p(\mu,\nu;P_{\theta_l}))}{\sum_{l'=1}^L f(\underline{W}_p^p(\mu,\nu;P_{\theta_{l'}}))}$ and $\theta_1, \ldots, \theta_L \sim \mathcal{U}(\mathbb{S}^{d-1})$.

Lower bounds. We summarize the connection between SW, Max-SW, EBSW, and Wasserstein distance in the following remark. The detail of the proof can be found in Nguyen & Ho (2023).

Remark 1. *Given any $\mu \in \mathcal{P}_p(\mathbb{R}^d)$ and $\nu \in \mathcal{P}_p(\mathbb{R}^d)$, we have:*

(a) $SW_p(\mu, \nu) \leq EBSW_p(\mu, \nu) \leq$ Max-$SW_p(\mu, \nu) \leq W_p(\mu, \nu)$,

(b) $\widehat{SW}_p(\mu, \nu; \theta_1, \ldots, \theta_L) \leq \widehat{EBSW}_p(\mu, \nu; \theta_1, \ldots, \theta_L) \leq W_p(\mu, \nu)$ for any $\theta_1, \ldots, \theta_L \in \mathbb{S}^{d-1}$,

(c) $\widehat{\text{Max-}SW}_p^p(\mu, \nu; \hat{\theta}_T) \leq W_p(\mu, \nu)$ for any $\hat{\theta}_T \in \mathbb{S}^{d-1}$.

Lifted sliced Wasserstein distances. Given $\mu \in \mathcal{P}_p(\mathbb{R}^d)$ and $\nu \in \mathcal{P}_p(\mathbb{R}^d)$, a lifted sliced Wasserstein-p distance can be defined as follows (Rowland et al., 2019):

$$LSW_p^p(\mu, \nu; \sigma) = \mathbb{E}_{\theta\sim\sigma}\left[\overline{W}_p^p(\mu, \nu; P_\theta)\right], \tag{6}$$

where $P_\theta : \mathbb{R}^d \to \mathbb{R}$ is the projection function, $\overline{W}_p^p(\mu, \nu; P_\theta)$ is the SWGG (equation 3), and $\sigma \in \mathcal{P}(\mathbb{S}^{d-1})$ is the slicing distribution. Similar to SW, we can obtain variants of PW by choosing σ. 1. *Fixed prior:* The original LSW is introduced as in projected Wasserstein (PW) in Rowland et al. (2019), which uses the uniform distribution $\mathcal{U}(\mathbb{S}^{d-1})$. 2. *Optimization-based:* In contrast to

the case of one-dimensional projected Wasserstein, which is always a lower bound of Wasserstein distance, SWGG is always an upper bound of Wasserstein distance. Therefore, it is desirable to select θ that can minimize the corresponding lifted cost, that leads to min SWGG distance: Min-$SWGG_p(\mu,\nu) = \min_{\theta\in\mathbb{S}^{d-1}} \overline{W}_p(\mu,\nu;P_\theta)$. 3. *Energy-based:* Similar to the case of EBSW, authors in Liu et al. (2025) proposes to choose σ as an energy-based distribution with the unnormalized density: $p_\sigma(\theta) \propto f(-\underline{W}_p^p(\mu,\nu;P_\theta))$, where f is often chosen to be an exponential function with temperature. The authors name the distance as expected sliced transport (EST).

Empirical estimation. For PW, Monte Carlo samples are used to approximate the distance: $\widehat{PW}_p^p(\mu,\nu;\theta_1,\ldots,\theta_L) = \frac{1}{L}\sum_{l=1}^L \overline{W}_p^p(\mu,\nu;P_{\theta_l})$, where $\theta_1,\ldots,\theta_L \overset{i.i.d}{\sim} \mathcal{U}(\mathbb{S}^{d-1})$. For Min-SWGG, we can use $\hat{\theta}_T$ which is the solution of an optimization algorithm with $T>0$ iterations, e.g., simulated annealing (Mahey et al., 2023), gradient ascent with a surrogate objective (Mahey et al., 2023), and differentiable approximation (Chapel et al., 2025): $\widehat{\text{Min-}SWGG}_p^p(\mu,\nu;\hat{\theta}_T) = \overline{W}_p^p(\mu,\nu;P_{\hat{\theta}_T})$. For EST, importance sampling estimation is used: $\widehat{EST}_p^p(\mu,\nu;\theta_1,\ldots,\theta_L)) = \sum_{l=1}^L \hat{w}_l \overline{W}_p^p(\mu,\nu;P_{\theta_l})$, where $\hat{w}_l = \frac{f(-\overline{W}_p^p(\mu,\nu;P_{\theta_l}))}{\sum_{l'=1}^L f(-\overline{W}_p^p(\mu,\nu;P_{\theta_{l'}}))}$ and $\theta_1,\ldots,\theta_L \sim \mathcal{U}(\mathbb{S}^{d-1})$.

Upper bounds. We summarize the connection between PW, Min-SWGG, EST, and Wasserstein distance in the following remark. The connection between Min-SWGG, EST, and Wasserstein distance is discussed in Mahey et al. (2023); Liu et al. (2025). The connection between EST and PW can be generalized from the connection between EBSW and SW in Nguyen & Ho (2023).

Remark 2. *Given any $\mu\in\mathcal{P}_p(\mathbb{R}^d)$ and $\nu\in\mathcal{P}_p(\mathbb{R}^d)$, we have:*

(a) $W_p(\mu,\nu) \le$ *Min-*$SWGG_p(\mu,\nu) \le EST_p(\mu,\nu) \le PW_p(\mu,\nu)$,

(b) $W_p(\mu,\nu) \le \widehat{EST}_p(\mu,\nu;\theta_1,\ldots,\theta_L) \le \widehat{PW}_p(\mu,\nu;\theta_1,\ldots,\theta_L)$ *for any* $\theta_1,\ldots,\theta_L\in\mathbb{S}^{d-1}$,

(c) $W_p(\mu,\nu) \le$ *Min-*$\widehat{SWGG}_p^p(\mu,\nu;\hat{\theta}_T)$ *for any* $\hat{\theta}_T\in\mathbb{S}^{d-1}$.

3.2 Regression of Wasserstein distance on sliced Wasserstein distances

We consider the setting where we observe pairs of distributions $(\mu_1,\nu_1),\ldots,(\mu_N,\nu_N)\sim\mathbb{P}(\mu,\nu)$. Here, $\mathbb{P}(\mu,\nu)$ is the meta distribution, and we are interested in relating $W_p(\mu_i,\nu_i)$ with $K>0$ SW distances $S_p^{(1)}(\mu_i,\nu_i),\ldots,S_p^{(K)}(\mu_i,\nu_i)$ for $i=1,\ldots,N$. We first start with a general model.

Definition 1 (Regression of Wasserstein distance onto SW distances)**.** *Given a meta distribution $\mathbb{P}(\mu,\nu)\in\mathcal{P}(\mathcal{P}_p(\mathbb{R}^d)\times\mathcal{P}_p(\mathbb{R}^d))$, $K>0$ SW distances $S_p^{(1)},\ldots,S_p^{(K)}$, a regression model of Wasserstein distance onto SW distances is defined as follows:*

$$W_p(\mu,\nu) = f(S_p^{(1)}(\mu,\nu),\ldots,S_p^{(K)}(\mu,\nu)) + \varepsilon, \tag{7}$$

where $(\mu,\nu)\sim\mathbb{P}(\mu,\nu)$, $f\in\mathcal{F}$ is the regression function, and ε is a noise model such that $\mathbb{E}[\varepsilon]=0$.

To estimate f, one natural estimator is the least square estimate:

$$f_{LSE} = \arg\min_{f\in\mathcal{F}} \mathbb{E}\left[\left(f(S_p^{(1)}(\mu,\nu),\ldots,S_p^{(K)}(\mu,\nu)) - W_p(\mu,\nu)\right)^2\right]. \tag{8}$$

It is worth noting that the function f can be constructed in both parametric ways (e.g., deep neural networks) or non-parametric ways (e.g., using kernels). However, in order to have a simple and explainable model, we consider linear functions in this work.

Linear Regression of Wasserstein distance onto SW distances. We now propose linear estimations of Wasserstein distances from SW distances.

Definition 2 (Linear Regression of Wasserstein distance onto SW distances)**.** *Given a meta distribution $\mathbb{P}(\mu,\nu)\in\mathcal{P}(\mathcal{P}_p(\mathbb{R}^d)\times\mathcal{P}_p(\mathbb{R}^d))$, $K>0$ SW distances $S_p^{(1)},\ldots,S_p^{(K)}$, the linear regression model of Wasserstein distance onto SW distances is defined as follows:*

$$W_p(\mu,\nu) = \sum_{k=1}^K \omega_k S_p^{(k)}(\mu,\nu) + \varepsilon, \tag{9}$$

where $(\mu,\nu)\sim\mathbb{P}(\mu,\nu)$ and ε is a noise model such that $\mathbb{E}[\varepsilon]=0$.

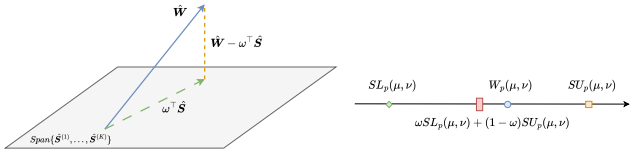

Figure 1: Linear regression of the Wasserstein distance vector $\hat{\boldsymbol{W}}$ on sliced Wasserstein (SW) distances $\hat{\boldsymbol{S}}^{(1)}, \ldots, \hat{\boldsymbol{S}}^{(K)}$. The left figure illustrates a linear model, interpreted as the $\mathbb{L}_2$ projection of the Wasserstein distance onto the linear span of the SW distances. The right figure depicts a special case of a constrained linear model with only two SW distances as predictors, which can be seen as a midpoint method.

Again, we use least-squares estimation to obtain an estimate of ω.

Remark 3. *The least square estimator admits the following closed form:*

$$\boldsymbol{\omega}_{LSE} = \mathbb{E}\left[\boldsymbol{S}_p(\mu,\nu)\boldsymbol{S}_p(\mu,\nu)^\top\right]^{-1} \mathbb{E}\left[\boldsymbol{S}_p(\mu,\nu) W_p(\mu,\nu)\right], \tag{10}$$

where $\boldsymbol{S}_p(\mu,\nu) = (S_p^{(1)}(\mu,\nu), \ldots, S_p^{(K)}(\mu,\nu))^\top$.

The detail of Remark 3 in given in Appendix A.1. In practice, we can sample $(\mu_1,\nu_1), \ldots, (\mu_M,\nu_M) \sim \mathbb{P}(\mu,\nu)$ to approximate the expectation in equation 10. Let $\hat{\boldsymbol{S}} \in \mathbb{R}_+^{M\times K}$ be the SW distances matrix i.e., $\hat{\boldsymbol{S}}_{ik} = S_p^{(k)}(\mu_i,\nu_i)$ for $i = 1,\ldots,M$, and $\hat{\boldsymbol{W}} \in \mathbb{R}_+^M$ be the Wasserstein distances vector i.e., $\hat{\boldsymbol{W}}_i = W_p(\mu_i,\nu_i)$ for $i = 1,\ldots,M$, we have the sample-based least-squares estimate: $\hat{\boldsymbol{\omega}}_{LSE} = (\hat{\boldsymbol{S}}^\top \hat{\boldsymbol{S}})^{-1} \hat{\boldsymbol{S}}^\top \hat{\boldsymbol{W}}$, which is an unbiased estimate of $\boldsymbol{\omega}$. It is well-known that the linear model can be seen as $\mathbb{L}_2$ projection of the Wasserstein distances vector $\hat{\boldsymbol{W}}$ onto the linear span of the SW distances vectors $\hat{\boldsymbol{S}}^{(1)}, \ldots, \hat{\boldsymbol{S}}^{(K)}$. We illustrate the idea in the left figure in Figure 1.

From Section 3.1, we know that SW distances are either lower bounds or upper bounds of Wasserstein distance. Therefore, natural estimation can be formed using midpoint method. In particular, given a lower bound $SL_p(\mu,\nu)$ and a upper bound $SU_p(\mu,\nu)$, we can predict the Wasserstein distance as $\omega_1 SL_p(\mu,\nu) + \omega_2 SU_p(\mu,\nu)$ with $0 \le \omega_1 \le 1$ and $\omega_2 = 1 - \omega_1$.

Definition 3 (Constrained Linear Regression of Wasserstein distance onto SW distances)**.** *Given a meta distribution* $\mathbb{P}(\mu,\nu) \in \mathcal{P}(\mathcal{P}_p(\mathbb{R}^d) \times \mathcal{P}_p(\mathbb{R}^d))$*,* $K > 0$ *SW distances* $SL_p^{(1)}, \ldots, SL_p^{(K)}$ *which are lower bounds of* W_p *and* $K > 0$ *SW distances* $SU_p^{(1)}, \ldots, SU_p^{(K)}$ *which are lower bounds of* W_p*, the constrained linear regression model is defined as follows:*

$$W_p(\mu,\nu) = \frac{1}{K}\sum_{k=1}^K \omega_k SL_p^{(k)}(\mu,\nu) + \frac{1}{K}\sum_{k=1}^K (1-\omega_k) SU_p^{(k)}(\mu,\nu) + \varepsilon, \tag{11}$$

where $0 \le \omega_k \le 1$*,* $(\mu,\nu) \sim \mathbb{P}(\mu,\nu)$ *and* ε *is a noise model such that* $\mathbb{E}[\varepsilon] = 0$*.*

To estimate $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_K)$ under the constrained model, we again form the least square estimate, which can be solved using quadratic programming and Monte Carlo estimation. In a special case where $K = 1$, i.e., having one lower bound and one upper bound, we can have a closed-form.

Remark 4. *For the case* $K = 1$ *with a lower bound* $SL_p(\mu,\nu)$ *and an upper bound* $SU_p(\mu,\nu)$*, a closed-form of the least square estimate under the constrained model can be formed:*

$$\hat{\omega}_{CLSE} = \frac{\mathbb{E}\left[(SU_p(\mu,\nu) - SL_p(\mu,\nu))(SU_p(\mu,\nu) - W_p(\mu,\nu))\right]}{\mathbb{E}[(SU_p(\mu,\nu) - SL_p(\mu,\nu)^2]}. \tag{12}$$

The detail of Remark 4 in given in Appendix A.2. The corresponding sample-based estimator for the model is: $\hat{\omega}_{CLSE} = \frac{\frac{1}{M}\sum_{i=1}^m (SU_p(\mu_i,\nu_i) - SL_p(\mu_i,\nu_i))(SU_p(\mu_i,\nu_i) - W_p(\mu_i,\nu_i))}{\frac{1}{M}\sum_{i=1}^M (SU_p(\mu_i,\nu_i) - SL_p(\mu_i,\nu_i)^2}$. We show the idea in the right figure in Figure 1. Compared to the unconstrained model, the constrained model has half of the parameters. In addition, it adds inductive bias to the model, which is often helpful when having limited observed samples.

Wasserstein Distance Estimation with Few-Shot Regression. We recall that we observe $(\mu_1,\nu_1), \ldots, (\mu_N,\nu_N) \sim \mathbb{P}(\mu,\nu)$ in practice. It is not computationally efficient to compute the

Table 1: k-NN accuracy on point-cloud classification on ShapeNetV2 dataset.

Methods	R^2	k=1	k=3	k=5	k=10	k=15
WD	–	83.6% ± 0.0%	83.5% ± 0.0%	84.2% ± 0.0%	82.9% ± 0.0%	79.2% ± 0.0%
RG-s	0.868 ± 0.02	82.1% ± 0.1%	81.7% ± 0.1%	80.8% ± 0.1%	79.4% ± 0.2%	75.5% ± 0.2%
RG-e	0.926 ± 0.04	82.5% ± 0.1%	82.2% ± 0.1%	80.9% ± 0.2%	79.6% ± 0.3%	75.7% ± 0.3%
RG-o	0.774 ± 0.38	65.1% ± 0.3%	67.7% ± 0.3%	67.6% ± 0.5%	66.7% ± 0.5%	66.0% ± 0.5%
RG-se	0.935 ± 0.02	82.5% ± 0.4%	82.2% ± 0.4%	82.6% ± 0.5%	81.9% ± 0.5%	76.5% ± 0.5%
RG-seo	0.937 ± 0.01	82.8% ± 0.4%	83.3% ± 0.5%	83.5% ± 0.7%	82.3% ± 0.7%	77.9% ± 0.7%

discussed least square estimates using all N pairs of distributions since those estimates require evaluation of Wasserstein distances. We then sample a subset $(\mu'_1, \nu'_1), \ldots, (\mu'_N, \nu'_M)$ from the original set with $M << N$. After obtaining an estimate $\hat{\boldsymbol{\omega}}$ from $(\mu'_1, \nu'_1), \ldots, (\mu'_N, \nu'_M)$, we can form estimations of the Wasserstein distances for other pairs and any new pair of distributions given their SW distances. -

Computational complexities. We assume that N pairs of distributions have the number of supports be at most n and in d dimensions. For fitting the estimate on M pairs, we need to compute MK SW distances (using L projecting directions) which costs $\mathcal{O}(MKLn(\log n + d))$ in time and M Wasserstein distances which costs $\mathcal{O}(Mn^2(n \log n + d))$. Computing the least square estimate has the time complexity of $\mathcal{O}(MK^2 + K^3)$. Then, we compute $(N - M)K$ SW distances which costs $\mathcal{O}((N-M)KLn(\log n+d))$ and predict $(N-M)$ Wasserstein distances which costs $\mathcal{O}((N-M)K)$. Total time complexity is $\mathcal{O}(NKLn(\log n+d)) + Mn^2(n \log n+d)) + MK^2 + K^3 + (N - M)K)$ compared to $\mathcal{O}(Nn^2(n \log n + d))$ of computing Wasserstein distances for all N pairs.

Extensions on regression. In this work, we focus on regressing the Wasserstein-p distance. If other ground metrics are used e.g., geodesic distances on manifolds, variants of SW distances such as spherical sliced Wasserstein distances (Bonet et al., 2023a; Tran et al., 2024; Quellmalz et al., 2023), hyperbolic sliced Wasserstein distances (Bonet et al., 2023b), sliced Wasserstein for distributions over positive definite matrices (Bonet et al., 2023c), and other non-linear variants of sliced Wasserstein (Bonet et al., 2025; Chapel et al., 2025; Tanguy et al., 2025; Kolouri et al., 2019). However, they might not be upper/lower bounds of the corresponding Wasserstein distances. Moreover, to incorporate uncertainty quantification, we can also perform Bayesian inference (Box & Tiao, 2011), e.g., putting a prior on the regression function.

4 Experiments

We define some specific model instances: *RG-o* uses Max-SW and Min-SWGG as predictors; *RG-s* uses SW and PWD as predictors; *RG-e* uses EBSW and EST as predictors. We also consider two extensions: *RG-se* combines SW, EBSW, PWD, and EST, and *RG-seo* combines all six variants. For each instance, we have a *constrained* version and an *unconstrained* version as discussed.

We evaluate our methods in five parts, each with a distinct goal. First, in Section 4.1, we test practical use via k-NN on ShapeNetV2, reporting accuracy under different metrics. Second, in Section 4.2, we benchmark *RG* variants against Wormhole across MNIST point clouds, ShapeNetV2, MERFISH Cell Niches Zhang et al. (2021), and scRNA-seq atlas Persad et al. (2023), reporting R^2/MSE/MAE in low-data regimes. Third, in Section 4.3, we combine our framework with Wormhole to introduce *RG-Wormhole*, a hybrid that matches Wormhole's performance while requiring far less training time. We compare training time under varying batch sizes and epochs, as well as embedding, reconstruction, barycenter, and interpolation quality. In Appendix B.1, we run Mixture of Gaussian simulations to verify that our methods approximate the true Wasserstein distance from low to high dimensions. In Appendix B.3, we visualize metric-induced geometry with UMAP McInnes et al. (2018). Throughout, N denotes the number of training-set sizes, and M_0 the number of samples drawn from the training set, yielding $M = \frac{M_0(M_0-1)}{2}$ pairs used to estimate *RG* coefficients.

4.1 Point Cloud Classification

We evaluate unconstrained *RG* variants over a classification task over 10-class ShapeNetV2 with 500 training samples (N=500) and estimate *RG* weights from 10 samples (M_0=10) drawn from the training set. The details of the experimental setting and full results are provided in Appendix B.2.

Table 2: Approximation quality of Wormhole and *RG* variants across four datasets under a training set size of 100 samples. Each cell reports R^2, MSE, and MA) with respect to the exact Wasserstein distance.

Methods	MNIST Point Cloud			ShapeNetV2			MERFISH			scRNA-seq		
	R^2	MSE	MAE	R^2	MSE	MAE	R^2	MSE	MAE	R^2	MSE	MAE
Wormhole	0.28	4.3×10^{-1}	5.1×10^{-1}	0.65	6.6×10^{-2}	1.8×10^{-1}	-3.6	8.0×10^{-4}	2.1×10^{-2}	0.04	7.0×10^{-3}	7.8×10^{-2}
RG-s (constr.)	0.84	8.9×10^{-2}	2.3×10^{-1}	0.88	2.0×10^{-2}	1.1×10^{-1}	0.91	1.6×10^{-5}	3.0×10^{-3}	1.00	3.7×10^{-5}	3.0×10^{-3}
RG-e (constr.)	0.86	8.7×10^{-2}	2.3×10^{-1}	0.90	1.7×10^{-2}	1.0×10^{-1}	0.92	1.3×10^{-5}	3.0×10^{-3}	1.00	1.3×10^{-5}	1.0×10^{-3}
RG-o (constr.)	0.77	1.4×10^{-1}	2.8×10^{-1}	0.66	5.2×10^{-2}	1.8×10^{-1}	0.75	4.8×10^{-5}	6.0×10^{-3}	0.99	6.1×10^{-5}	6.0×10^{-3}
RG-se (constr.)	0.84	9.8×10^{-2}	2.4×10^{-1}	0.92	1.4×10^{-2}	9.3×10^{-2}	0.91	1.5×10^{-5}	3.0×10^{-3}	1.00	2.4×10^{-5}	2.0×10^{-3}
RG-seo (constr.)	0.85	9.0×10^{-2}	2.3×10^{-1}	0.91	1.7×10^{-2}	1.0×10^{-1}	0.92	1.3×10^{-5}	3.0×10^{-3}	1.00	2.2×10^{-5}	2.0×10^{-3}
RG-s (unconstr.)	0.93	4.5×10^{-2}	1.6×10^{-1}	0.94	1.1×10^{-2}	8.2×10^{-2}	0.96	6.3×10^{-6}	2.0×10^{-3}	0.99	8.6×10^{-5}	7.0×10^{-3}
RG-e (unconstr.)	0.92	5.4×10^{-2}	1.8×10^{-1}	0.92	1.5×10^{-2}	9.8×10^{-2}	0.96	6.9×10^{-6}	2.0×10^{-3}	0.99	7.0×10^{-5}	6.0×10^{-3}
RG-o (unconstr.)	0.77	1.4×10^{-1}	3.0×10^{-1}	0.75	3.8×10^{-2}	1.6×10^{-1}	0.89	8.7×10^{-4}	2.9×10^{-2}	0.82	2.9×10^{-3}	5.2×10^{-2}
RG-se (unconstr.)	0.93	4.0×10^{-2}	1.5×10^{-1}	0.95	9.9×10^{-3}	7.8×10^{-2}	0.98	2.9×10^{-6}	1.0×10^{-3}	1.00	3.0×10^{-5}	4.0×10^{-3}
RG-seo (unconstr.)	0.93	4.0×10^{-2}	1.5×10^{-1}	0.95	9.8×10^{-3}	7.8×10^{-2}	0.97	6.8×10^{-6}	2.0×10^{-3}	0.99	6.8×10^{-5}	7.0×10^{-3}

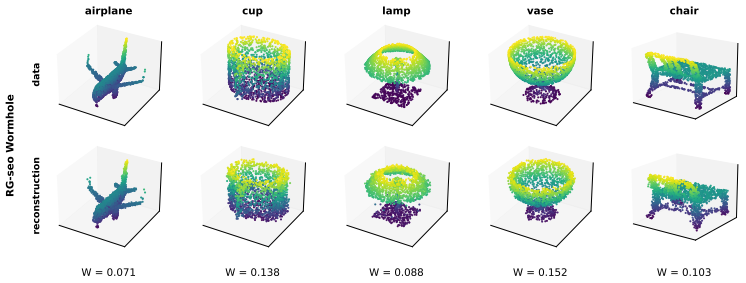

Figure 2: ModelNet40: a *RG-Wormhole* variant in reconstruction experiment.

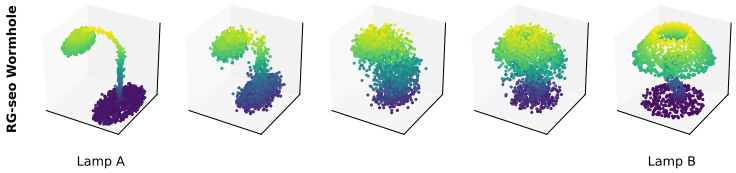

Figure 3: ModelNet40: a *RG-Wormhole* variant in interpolation experiment.

Results. Table 1 reports k-NN accuracy on ShapeNetV2 under different metrics. As expected, WD achieves the best accuracy, with 84.2% at k=5. Among single sliced-based metrics, SW and EBSW, are the strongest, though they cap at about 72.5% top-1. Our *RG* methods close much of the gap to Wasserstein. Both *RG-s* and *RG-e* consistently achieve around 82.5% top-1 accuracy with high correlation to Wasserstein ($R^2 \approx 0.9$). The multi-metric extensions further improve stability: *RG-se* and *RG-seo* reach up to 83.5% accuracy with R^2 as high as 0.93, essentially matching Wasserstein.

4.2 Comparisons of RG variants vs. Wormhole in low-data regimes

We compare our *RG* framework with Wormhole within the same training sizes, matching the preprocessing of (Haviv et al., 2024) across four datasets spanning dimensionality: MNIST pixel point clouds (2D), ShapeNetV2 point clouds (3D), MERFISH Niche Cells (254D), and scRNA-seq (2,500D). We train on $N \in \{10, 50, 100, 200\}$ random pairs and evaluate R^2/MSE/MAE against exact WD. For fairness, the number of training pairs for Wormhole equals the number used to estimate the linear coefficients for *RG* variants, i.e., M_0=N. Full results appear in Figures 6–13 with settings in Appendix B.4; Table 2 summarizes the M_0=100 case, and other M_0 follow the same pattern.

Results. Across all four datasets, *RG* variants consistently outperform Wormhole at small training sizes. Wormhole is weaker primarily because it is data hungry and its performance improves as we add samples, yet under comparable budgets it still trails our methods. By contrast, *RG* variants are already accurate with few pairs, with *unconstrained* variants are slightly stronger, whereas *constrained* variants converge faster and are preferable at the very smallest sizes. *RG-se* and *RG-seo* are the strongest when given sufficient samples though the latter can lag at the tiniest sizes before its weights settle but becomes top-performing quickly and still requires far fewer pairs than Wormhole.

4.3 RG-WORMHOLE: ACCELERATING WORMHOLE WITH REGRESSION OF WASSERSTEIN

The previous comparison reveals a clear trade-off. *RG* framework is lightweight and data-efficient, but it does not produce Euclidean embeddings and therefore cannot support interpolation experiments. Wormhole, in contrast, learns Euclidean embeddings that enable interpolation and reconstruction, but it is computationally heavy because training requires many Wasserstein evaluations (pairwise distances within each mini-batch and reconstruction losses), which slows and raises training cost.

RG-Wormhole. To combine the strengths of both, we introduce *RG-Wormhole*. We first calibrate a *RG* surrogate on a small set of exact Wasserstein pairs from the same data domain and freeze its weights. We then keep the Wormhole architecture, optimizer, and schedule unchanged, and simply replace every use of the Wasserstein distance with the calibrated surrogate in both the encoder (pairwise distances in the batch) and the decoder (reconstruction loss). No other component is modified. This substitution makes each training step much faster while preserving the performance.

We run five experiments of both models to empirically show that *RG-Wormhole* is much faster than Wormhole while keeping similar effectiveness. First, we measure training time by training Wormhole and *RG-Wormhole* under the same optimizer and schedule, sweeping batch sizes 4–20 and reporting wall-clock time for training-set sizes $N \in \{10, 50, 100, 200\}$. Second, we assess encoders via R^2/MSE/MAE between learned pairwise distances and exact Wasserstein. Third, we evaluate decoders via the Wasserstein loss between each input shape and its reconstruction. Fourth, we examine barycenters by decoding each class's mean embedding and visualizing results. Finally, we study interpolation by decoding linear paths between two embeddings and visualizing the trajectories. Across all experiments, hyperparameters match Wormhole; the only change in *RG-Wormhole* is replacing every use of the Wasserstein distance in the encoder and decoder losses with the calibrated unconstrained *RG* variants. For *RG-Wormhole*, we estimate the *RG* coefficient using 10 random training samples (M_0=10) before plugging into Wormhole. We provide some results in Figures2–3 though the details of experimental settings and full results can be found in Appendix B.5.

Results. Replacing every Wasserstein call in Wormhole with a calibrated *RG* variants preserves performance while cutting compute. First, in the training-time comparison (Figure 14 in Appendix B.5), *RG-Wormhole* is far faster than Wormhole across all batch sizes and training budgets, with a very large gap. As batch size increases, Wormhole's time grows almost exponentially, while *RG-Wormhole* rises only slightly, close to linear or even flat. Next, we verify that the trained models have similar quality. For the encoder, Figures 15 and 16 in Appendix B.5 show pairwise distances that align with the ground-truth Wasserstein and embeddings that match Wormhole, with essentially identical R^2, MSE, and MAE. For the decoder, Figures 17 and 18 in Appendix B.5 evaluate reconstructions against the original point clouds using the Wasserstein distance, and both *RG-Wormhole* and Wormhole produce very small and nearly identical distances. Finally we test whether *RG-Wormhole* preserves the geometry needed for downstream use. The decoded class barycenters from *RG-Wormhole* are clean and class consistent and they match those from Wormhole, we refer to Figure19 in Appendix B.5. We also interpolate by moving linearly in the embedding space and decoding along the path, and the trajectories from *RG-Wormhole* are smooth and semantically meaningful with no visible artifacts, we refer to Figure20 in Appendix B.5. Overall *RG-Wormhole* matches Wormhole while training much faster, which makes it a practical choice when compute is limited.

5 CONCLUSIONS

We introduced a regression framework mapping Wasserstein to sliced Wasserstein distances under a meta-distribution of random distribution pairs. Two simple linear models enable lightweight estimation, leading to the *RG* framework for few-shot Wasserstein approximation. We derived constrained and unconstrained forms and validated them through Mixture of Gaussian simulations, point cloud classification, and metric-space visualizations, where the surrogate closely matched the exact distance. Compared to Wormhole on MNIST, ShapeNetV2, MERFISH, and scRNA-seq, our method achieved better performance in low-data regimes. Replacing Wasserstein calls in Wormhole with our method yielded *RG-Wormhole*, preserving accuracy while greatly reducing training time.

REFERENCES

Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *International conference on machine learning*, pp. 40–49. PMLR, 2018.

David Alvarez-Melis and Nicolo Fusi. Geometric dataset distances via optimal transport. *Advances in Neural Information Processing Systems*, 33:21428–21439, 2020.

Clément Bonet, Paul Berg, Nicolas Courty, François Septier, Lucas Drumetz, and Minh-Tan Pham. Spherical sliced-Wasserstein. *International Conference on Learning Representations*, 2023a.

Clément Bonet, Laetitia Chapel, Lucas Drumetz, and Nicolas Courty. Hyperbolic sliced-Wasserstein via geodesic and horospherical projections. In *Topological, Algebraic and Geometric Learning Workshops 2023*, pp. 334–370. PMLR, 2023b.

Clément Bonet, Benoıt Malézieux, Alain Rakotomamonjy, Lucas Drumetz, Thomas Moreau, Matthieu Kowalski, and Nicolas Courty. Sliced-Wasserstein on symmetric positive definite matrices for m/eeg signals. In *International Conference on Machine Learning*, pp. 2777–2805. PMLR, 2023c.

Clément Bonet, Lucas Drumetz, and Nicolas Courty. Sliced-wasserstein distances and flows on cartan-hadamard manifolds. *Journal of Machine Learning Research*, 26(32):1–76, 2025.

Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and Radon Wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 1(51):22–45, 2015.

George EP Box and George C Tiao. *Bayesian inference in statistical analysis*. John Wiley & Sons, 2011.

Charlotte Bunne, Stefan G Stark, Gabriele Gut, Jacobo Sarabia Del Castillo, Mitch Levesque, Kjong-Van Lehmann, Lucas Pelkmans, Andreas Krause, and Gunnar Rätsch. Learning single-cell perturbation responses using neural optimal transport. *Nature methods*, 20(11):1759–1768, 2023.

Laetitia Chapel, Romain Tavenard, and Samuel Vaiter. Differentiable generalized sliced Wasserstein plans. *arXiv preprint arXiv:2505.22049*, 2025.

Yaqing Chen, Zhenhua Lin, and Hans-Georg Müller. Wasserstein regression. *Journal of the American Statistical Association*, 118(542):869–882, 2023.

Nicolas Courty, Rémi Flamary, and Mélanie Ducoffe. Learning Wasserstein embeddings. In *International Conference on Learning Representations*, 2018.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in Neural Information Processing Systems*, pp. 2292–2300, 2013.

Ishan Deshpande, Yuan-Ting Hu, Ruoyu Sun, Ayis Pyrros, Nasir Siddiqui, Sanmi Koyejo, Zhizhen Zhao, David Forsyth, and Alexander G Schwing. Max-sliced Wasserstein distance and its use for GANs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 10648–10656, 2019.

DC Dowson and BV666017 Landau. The fréchet distance between multivariate normal distributions. *Journal of multivariate analysis*, 12(3):450–455, 1982.

Jean Feydy, Benjamin Charlier, François-Xavier Vialard, and Gabriel Peyré. Optimal transport for diffeomorphic registration. In *Medical Image Computing and Computer Assisted Intervention-MICCAI 2017: 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part I 20*, pp. 291–299. Springer, 2017.

Aude Genevay, Gabriel Peyré, and Marco Cuturi. Learning generative models with Sinkhorn divergences. In *International Conference on Artificial Intelligence and Statistics*, pp. 1608–1617. PMLR, 2018.

Doron Haviv, Russell Zhang Kunes, Thomas Dougherty, Cassandra Burdziak, Tal Nawy, Anna Gilbert, and Dana Pe'er. Wasserstein wormhole: Scalable optimal transport distance with Transformer. In *Forty-first International Conference on Machine Learning*, 2024.

Soheil Kolouri, Kimia Nadjahi, Umut Simsekli, Roland Badeau, and Gustavo Rohde. Generalized sliced Wasserstein distances. In *Advances in Neural Information Processing Systems*, pp. 261–272, 2019.

Soheil Kolouri, Navid Naderializadeh, Gustavo K. Rohde, and Heiko Hoffmann. Wasserstein embedding for graph learning. In *International Conference on Learning Representations*, 2021.

Tianyi Lin, Chenyou Fan, Nhat Ho, Marco Cuturi, and Michael Jordan. Projection robust Wasserstein distance and Riemannian optimization. *Advances in Neural Information Processing Systems*, 33: 9383–9397, 2020.

Xinran Liu, Rocio Diaz Martin, Yikun Bai, Ashkan Shahbazi, Matthew Thorpe, Akram Aldroubi, and Soheil Kolouri. Expected sliced transport plans. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=P7O1Vt1BdU.

Guillaume Mahey, Laetitia Chapel, Gilles Gasso, Clément Bonet, and Nicolas Courty. Fast optimal transport through sliced generalized wasserstein geodesics. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 35350–35385, 2023.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Boris Muzellec and Marco Cuturi. Subspace detours: Building transport plans that are optimal on subspace projections. In *Advances in Neural Information Processing Systems*, pp. 6917–6928, 2019.

Khai Nguyen. An introduction to sliced optimal transport. *arXiv preprint arXiv:2508.12519*, 2025.

Khai Nguyen and Nhat Ho. Energy-based sliced Wasserstein distance. *Advances in Neural Information Processing Systems*, 2023.

Khai Nguyen and Nhat Ho. Sliced Wasserstein estimator with control variates. *International Conference on Learning Representations*, 2024.

Khai Nguyen, Nhat Ho, Tung Pham, and Hung Bui. Distributional sliced-Wasserstein and applications to generative modeling. In *International Conference on Learning Representations*, 2021.

Khai Nguyen, Nicola Bariletto, and Nhat Ho. Quasi-monte carlo for 3d sliced Wasserstein. In *International Conference on Learning Representations*, 2024.

Sloan Nietert, Ziv Goldfeld, Ritwik Sadhu, and Kengo Kato. Statistical, robustness, and computational guarantees for sliced wasserstein distances. *Advances in Neural Information Processing Systems*, 35:28179–28193, 2022.

Sitara Persad, Zi-Ning Choo, Christine Dien, Noor Sohail, Ignas Masilionis, Ronan Chaligné, Tal Nawy, Chrysothemis C Brown, Roshan Sharma, Itsik Pe'er, et al. Seacells infers transcriptional and epigenomic cellular states from single-cell genomics data. *Nature biotechnology*, 41(12): 1746–1757, 2023.

Gabriel Peyré and Marco Cuturi. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

Michael Quellmalz, Robert Beinert, and Gabriele Steidl. Sliced optimal transport on the sphere. *Inverse Problems*, 39(10):105005, 2023.

Julien Rabin, Julie Delon, and Yann Gousseau. Regularization of transportation maps for color and contrast transfer. In *2010 IEEE International Conference on Image Processing*, pp. 1933–1936. IEEE, 2010.

Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot. Wasserstein barycenter and its application to texture mixing. In *Scale Space and Variational Methods in Computer Vision: Third International Conference, SSVM 2011, Ein-Gedi, Israel, May 29–June 2, 2011, Revised Selected Papers 3*, pp. 435–446. Springer, 2012.

Mark Rowland, Jiri Hron, Yunhao Tang, Krzysztof Choromanski, Tamas Sarlos, and Adrian Weller. Orthogonal estimation of Wasserstein distances. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 186–195. PMLR, 2019.

Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. A metric for distributions with applications to image databases. In *Sixth international conference on computer vision (IEEE Cat. No. 98CH36271)*, pp. 59–66. IEEE, 1998.

Meyer Scetbon, Marco Cuturi, and Gabriel Peyré. Low-rank sinkhorn factorization. In *International Conference on Machine Learning*, pp. 9344–9354. PMLR, 2021.

Keanu Sisouk, Julie Delon, and Julien Tierny. A user's guide to sampling strategies for sliced optimal transport. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856.

Eloi Tanguy. Convergence of sgd for training neural networks with sliced Wasserstein losses. *arXiv preprint arXiv:2307.11714*, 2023.

Eloi Tanguy, Laetitia Chapel, and Julie Delon. Sliced optimal transport plans. *arXiv preprint arXiv:2508.01243*, 2025.

Huy Tran, Yikun Bai, Abihith Kothapalli, Ashkan Shahbazi, Xinran Liu, Rocio Diaz Martin, and Soheil Kolouri. Stereographic spherical sliced Wasserstein distances. *International Conference on Machine Learning*, 2024.

Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

Cédric Villani. *Optimal transport: old and new*, volume 338. Springer, 2009.

Fang Wu, Nicolas Courty, Shuting Jin, and Stan Z Li. Improving molecular representation learning with metric learning-enhanced optimal transport. *Patterns*, 4(4), 2023.

Meng Zhang, Stephen W Eichhorn, Brian Zingg, Zizhen Yao, Kaelan Cotter, Hongkui Zeng, Hongwei Dong, and Xiaowei Zhuang. Spatially resolved cell atlas of the mouse primary motor cortex by merfish. *Nature*, 598(7879):137–143, 2021.

Supplement to "Fast Estimation of Wasserstein Distances via Regression on Sliced Wasserstein Distances"

A Details

A.1 Details of Remark 3

We derive the gradient:

$$\begin{aligned}
&\nabla_{\boldsymbol{\omega}}\mathbb{E}\left[\left\|\boldsymbol{\omega}^\top \boldsymbol{S}_p^{(k)}(\mu,\nu) - W_p(\mu,\nu))\right\|_2^2\right] \\
&= \nabla_{\boldsymbol{\omega}}\mathbb{E}\left[(\boldsymbol{\omega}^\top \boldsymbol{S}_p^{(k)}(\mu,\nu) - W_p(\mu,\nu)))^\top(\boldsymbol{\omega}^\top \boldsymbol{S}_p^{(k)}(\mu,\nu) - W_p(\mu,\nu)))\right] \\
&= \nabla_{\boldsymbol{\omega}}\mathbb{E}\left[\boldsymbol{\omega}^\top \boldsymbol{S}_p^{(k)}(\mu,\nu)\boldsymbol{S}_p^{(k)}(\mu,\nu)^\top\boldsymbol{\omega}\right] - 2\nabla_{\boldsymbol{\omega}}\mathbb{E}\left[\boldsymbol{S}_p^{(k)}(\mu,\nu)^\top\boldsymbol{\omega}W_p(\mu,\nu)\right] \quad (13)\\
&= \mathbb{E}\left[\nabla_{\boldsymbol{\omega}}\boldsymbol{\omega}^\top \boldsymbol{S}_p^{(k)}(\mu,\nu)\boldsymbol{S}_p^{(k)}(\mu,\nu)^\top\boldsymbol{\omega}\right] - 2\mathbb{E}\left[\nabla_{\boldsymbol{\omega}}\boldsymbol{S}_p^{(k)}(\mu,\nu)^\top\boldsymbol{\omega}W_p(\mu,\nu)\right] \quad (14)\\
&= 2\mathbb{E}\left[\boldsymbol{S}_p^{(k)}(\mu,\nu)\boldsymbol{S}_p^{(k)}(\mu,\nu)^\top\right]\boldsymbol{\omega} - 2\mathbb{E}\left[\boldsymbol{S}_p^{(k)}(\mu,\nu)W_p(\mu,\nu)\right] \quad (15)
\end{aligned}$$

Setting the gradient to 0, we obtain

$$\hat{\boldsymbol{\omega}}_{LSE} = \mathbb{E}\left[\boldsymbol{S}_p^{(k)}(\mu,\nu)\boldsymbol{S}_p^{(k)}(\mu,\nu)^\top\right]^{-1}\mathbb{E}\left[\boldsymbol{S}_p^{(k)}(\mu,\nu)W_p(\mu,\nu)\right], \quad (16)$$

which completes the proof.

A.2 Details of Remark 4

From the definition, we recall the model:

$$W_p(\mu,\nu) = \sum_{k=1}^{K}\omega_k SL_p^{(k)}(\mu,\nu) + \sum_{k=1}^{K}(1-\omega_k)SU_p^{(k)}(\mu,\nu) + \varepsilon. \quad (17)$$

With $K = 1$, we rewrite the model as follows:

$$W_p(\mu,\nu) = \omega SL_p(\mu,\nu) + (1-\omega)SU_p(\mu,\nu) + \varepsilon, \quad (18)$$

which is equivalent to

$$W_p(\mu,\nu) - SU_p(\mu,\nu) = \omega(SL_p(\mu,\nu) - SU_p(\mu,\nu)) + \epsilon. \quad (19)$$

Since equation 19 is again an unconstrained linear model, we can obtain the least-squares estimate by following Appendix A.1:

$$\hat{\omega}_{CLSE} = \frac{\mathbb{E}\left[(SU_p(\mu,\nu) - SL_p(\mu,\nu))(SU_p(\mu,\nu) - W_p(\mu,\nu))\right]}{\mathbb{E}[(SU_p(\mu,\nu) - SL_p(\mu,\nu)^2]}, \quad (20)$$

which concludes the proof.

B Experiments

B.1 Gaussian Simulation

We study how a lower–upper bound pair approximates the Wasserstein distance as dimension grows. We simulate 3-component Gaussian mixtures for d=1 . . . 100 (10 seeds), with 200 points per component. For each pair we compute the exact Wasserstein and six sliced-based metrics. Focusing on *RG-o*, *RG-s*, and *RG-e*, we fit a constrained weight $w \in [0,1]$ and report the estimated weight $\hat{w}$ and R^2 versus the exact Wasserstein.

Results. We refer to Figure 4 for the result. The fits are strong for all three methods and all dimensions: R^2 is always above 0.8 and quickly rises to $\approx$ 0.9-1.0. We also see a clear pattern in the

weights: as dimension grows, the weight on the lower bound goes down, so the upper-bound metric gets more weight and eventually dominates. In short, high dimensions favor the upper bound, while lower dimensions rely more on the lower bound.

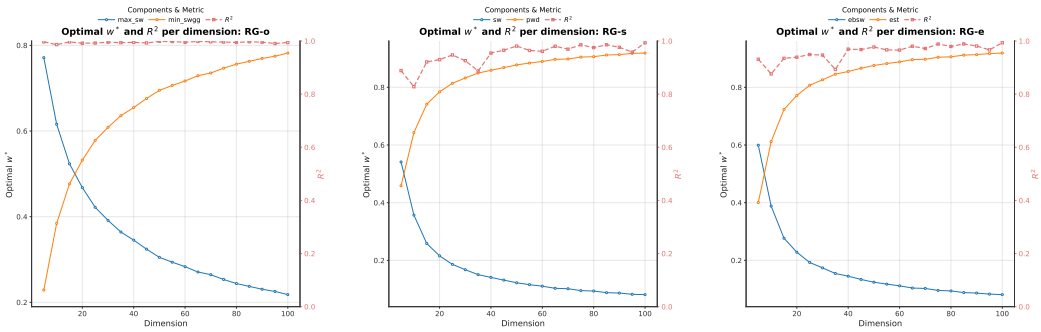

Figure 4: Optimal w^* and R^2 in each dimension

B.2 Point Cloud Classification

Experimental settings. We construct a 10-class subset, centralize, normalize each shape so that all coordinates lie in $[-1, 1]^3$, and uniformly subsample 2,048 points per shape. For each class we select 50 training examples and 100 test examples. We then compute pairwise distance matrices between train and test sets under different metrics, and evaluate classification accuracy using a k-nearest neighbor classifier with $k \in \{1, 3, 5, 10, 15\}$. Besides the six individual sliced-based metrics, we include all *RG* variants in unconstrained version. We use 10 samples drawn from the training set to estimate the linear coefficient of RG variants.

Table 3: k-NN accuracy on point-cloud classification on ShapeNetV2 dataset.

Methods	R^2	k=1	k=3	k=5	k=10	k=15
WD	–	83.6% ± 0.0%	83.5% ± 0.0%	84.2% ± 0.0%	82.9% ± 0.0%	79.2% ± 0.0%
SWD	–	72.4% ± 0.0%	71.4% ± 0.0%	70.4% ± 0.0%	69.0% ± 0.0%	66.7% ± 0.0%
PWD	–	42.6% ± 0.0%	42.9% ± 0.0%	40.4% ± 0.0%	39.3% ± 0.0%	39.0% ± 0.0%
EBSW	–	72.5% ± 0.0%	69.2% ± 0.0%	60.4% ± 0.0%	67.9% ± 0.0%	65.3% ± 0.0%
EST	–	39.1% ± 0.0%	40.4% ± 0.0%	40.2% ± 0.0%	38.0% ± 0.0%	36.5% ± 0.0%
Max-SW	–	60.3% ± 0.0%	54.6% ± 0.0%	57.7% ± 0.0%	57.6% ± 0.0%	56.8% ± 0.0%
Min-SWGG	–	36.4% ± 0.0%	37.6% ± 0.0%	35.0% ± 0.0%	32.9% ± 0.0%	30.8% ± 0.0%
RG-s	0.868 ± 0.02	82.1% ± 0.1%	81.7% ± 0.1%	80.8% ± 0.1%	79.4% ± 0.2%	75.5% ± 0.2%
RG-e	0.926 ± 0.04	82.5% ± 0.1%	82.2% ± 0.1%	80.9% ± 0.2%	79.6% ± 0.3%	75.7% ± 0.3%
RG-o	0.774 ± 0.38	65.1% ± 0.3%	67.7% ± 0.3%	67.6% ± 0.5%	66.7% ± 0.5%	66.0% ± 0.5%
RG-se	0.935 ± 0.02	82.5% ± 0.4%	82.2% ± 0.4%	82.6% ± 0.5%	81.9% ± 0.5%	76.5% ± 0.5%
RG-seo	0.937 ± 0.01	82.8% ± 0.4%	83.3% ± 0.5%	83.5% ± 0.7%	82.3% ± 0.7%	77.9% ± 0.7%

B.3 Metric Space Visualization

Experimental settings. We visualize the geometry each metric induces on ShapeNetV2. From 10 categories, we randomly sample 500 shapes per class, normalize each shape so that all coordinates lie in $[-1, 1]^3$, and keep 2,048 points per shape. For every method, we compute the pairwise distance matrix, then feed to UMAP to obtain 2D embeddings. We use 10 samples drawn from the training set to estimate the linear coefficient of RG variants.

Results. The result is visual in Figures 5. Across methods, the true Wasserstein produces well-separated class clusters with clear margins. The *RG* variants produce embeddings that are visually very close to the Wasserstein embeddings, preserving both local compactness and the global arrangement of classes. By contrast, single sliced baselines are weaker. SWD and EBSW keep some structure but blur boundaries, while Max-SW and Min-SWGG show more mixing and noise.

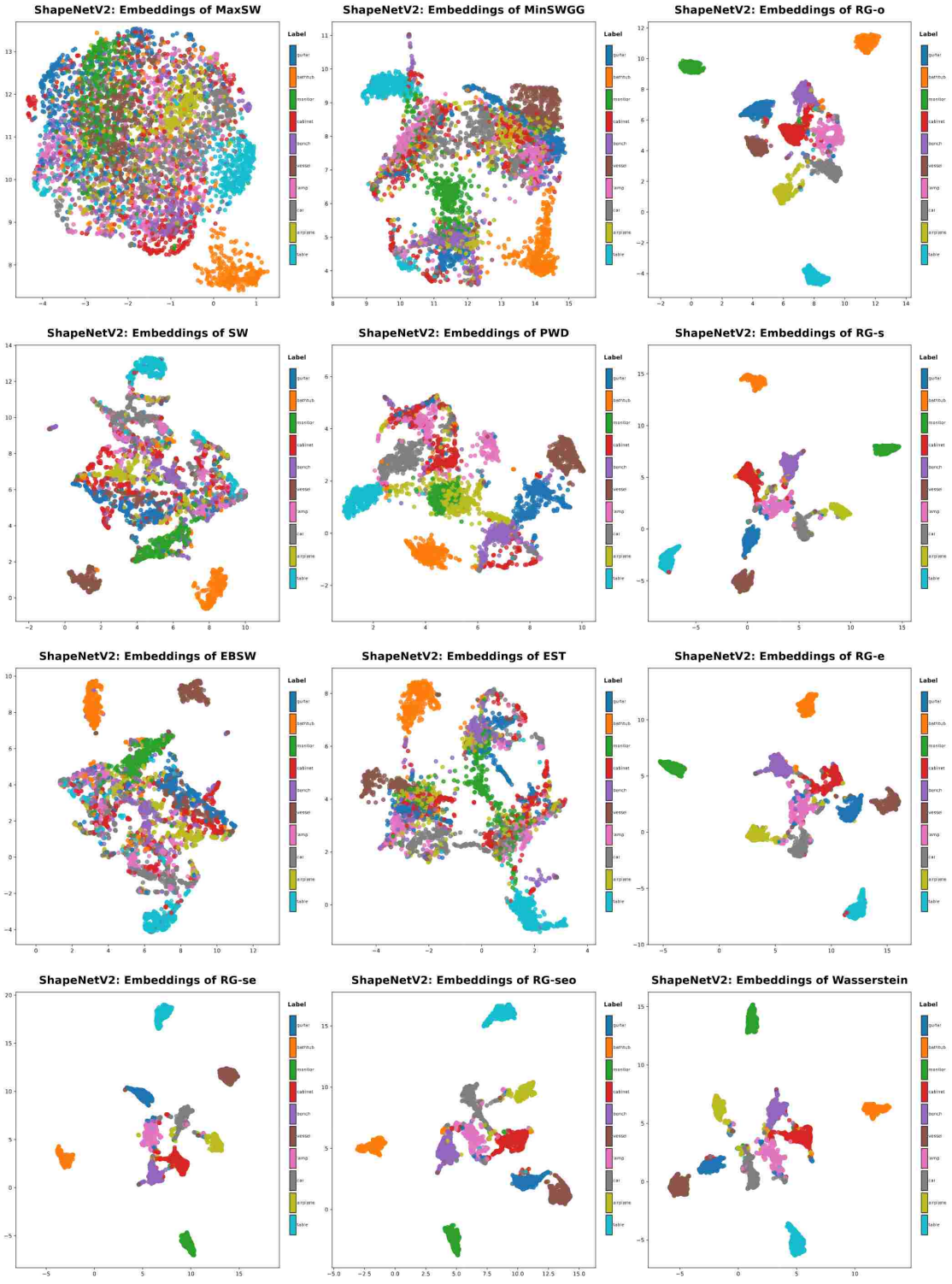

Figure 5: Embeddings of methods in ShapeNetV2 dataset.

B.4 Comparison of RG variants vs. Wormhole in low-data regimes

Experimental Settings. We compare our proposed *RG* framework against Wormhole, a state-of-the-art Wasserstein approximation method. To ensure fairness, we follow the exact preprocessing protocol of Haviv et al. (2024). We consider four datasets spanning a wide range of dimensionalities: (i) MNIST point clouds, obtained by thresholding 28×28 grayscale images and treating the active pixels as 2D point coordinates; (ii) ShapeNetV2 point clouds, where each CAD model is uniformly sampled into $2{,}048$ points in 3D and normalized; (iii) MERFISH Cell Niches, where each cell is represented by the 50 μm neighborhood of its gene-expression profile embedded in a 254-dimensional space; and (iv) scRNA-seq atlas data, where cells are aggregated into MetaCells that form $2{,}500$-dimensional gene-expression point clouds. We vary the number of training pairs $N \in \{10, 50, 100, 200\}$ by drawing pairs uniformly, and evaluate on $10{,}000$ independently sampled test pairs. For each dataset and training size, we report R^2, MSE, and MAE with respect to the exact Wasserstein.

The original Wormhole codebase is built on JAX and TensorFlow, which are not compatible with our environment. Accordingly, we reimplemented Wormhole in PyTorch.

Data Preprocessing. We follow the same preprocessing pipeline as Haviv et al. (2024).

- **MNIST Point Clouds.** We turn MNIST 28×28 images into 2D point clouds by thresholding pixel values at 0.5 and keeping the coordinates of the active pixels.
- **ShapeNetV2 Point Clouds.** We use ShapeNetCore.v2 with 15k points per shape. Each shape is normalized to fit inside a unit cube with coordinates in $[-1, 1]^3$. We then split each shape into 10k training points and 5k test points, and randomly sample $2{,}048$ points from each point cloud.
- **MERFISH Cell Niches.** We scale each gene's expression to $[-1, 1]$ and divide by $\sqrt{d}$, where d is the number of genes. For each cell, we use spatial positions to find its 11 nearest neighbors within a 50 μm radius, keeping only cells with enough neighbors with its cell-type label.
- **scRNA-seq.** We select $2{,}500$ highly variable genes, normalize counts (library-size 10^4 and $\log(1+x)$), and scale each gene to $[-1, 1]$ divided by $\sqrt{d}$ (d=2500). We then cluster cells with K-means. For each cluster seed, we consider it as a cloud, labeled by the seed's annotation.

Wormhole training hyperparameters. We follow the Transformer autoencoder setup of *Wormhole* with the configuration below:

Table 4: Wormhole training hyperparameters.

Component	Setting
Batch size	`10`
Optimizer / LR	Adam, `lr` $= 10^{-4}$
LR schedule	ExponentialLR, final factor ≈ 0.1 over all epochs
Epochs	2,000 epochs (20,000 steps)
Transformer depth	`num_layers` $= 3$
Attention heads	`num_heads` $= 4$
Embedding dim	`emb_dim` $= 128$
MLP hidden dim	`mlp_dim` $= 512$
Attention dropout	`attention_dropout_rate` $= 0.1$
Decoder coeff.	`coeff_dec` $= 0.1$

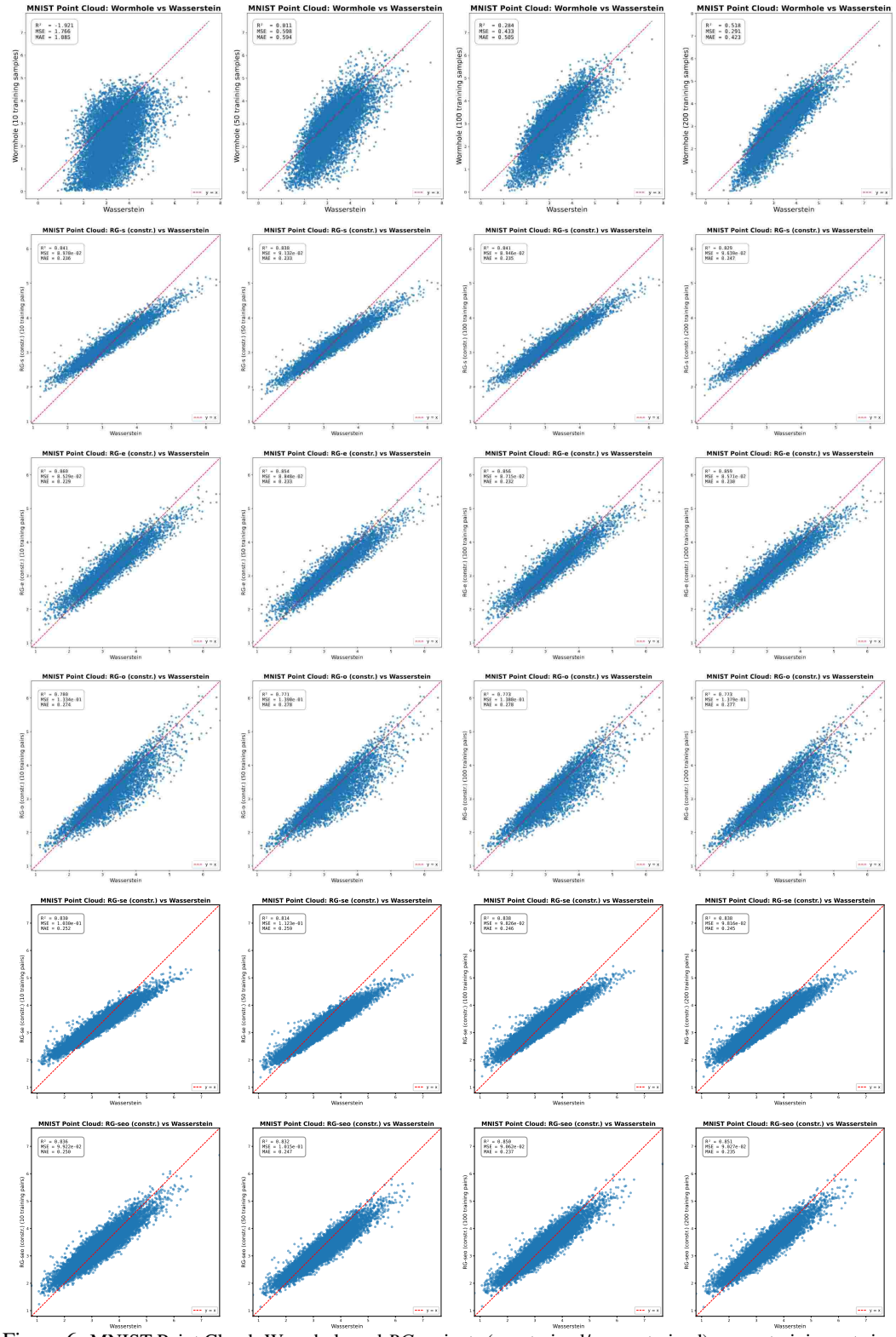

Figure 6: MNIST Point Cloud: Wormhole and *RG* variants (constrained/unconstrained) across training set sizes of 10, 50, 100 and 200.

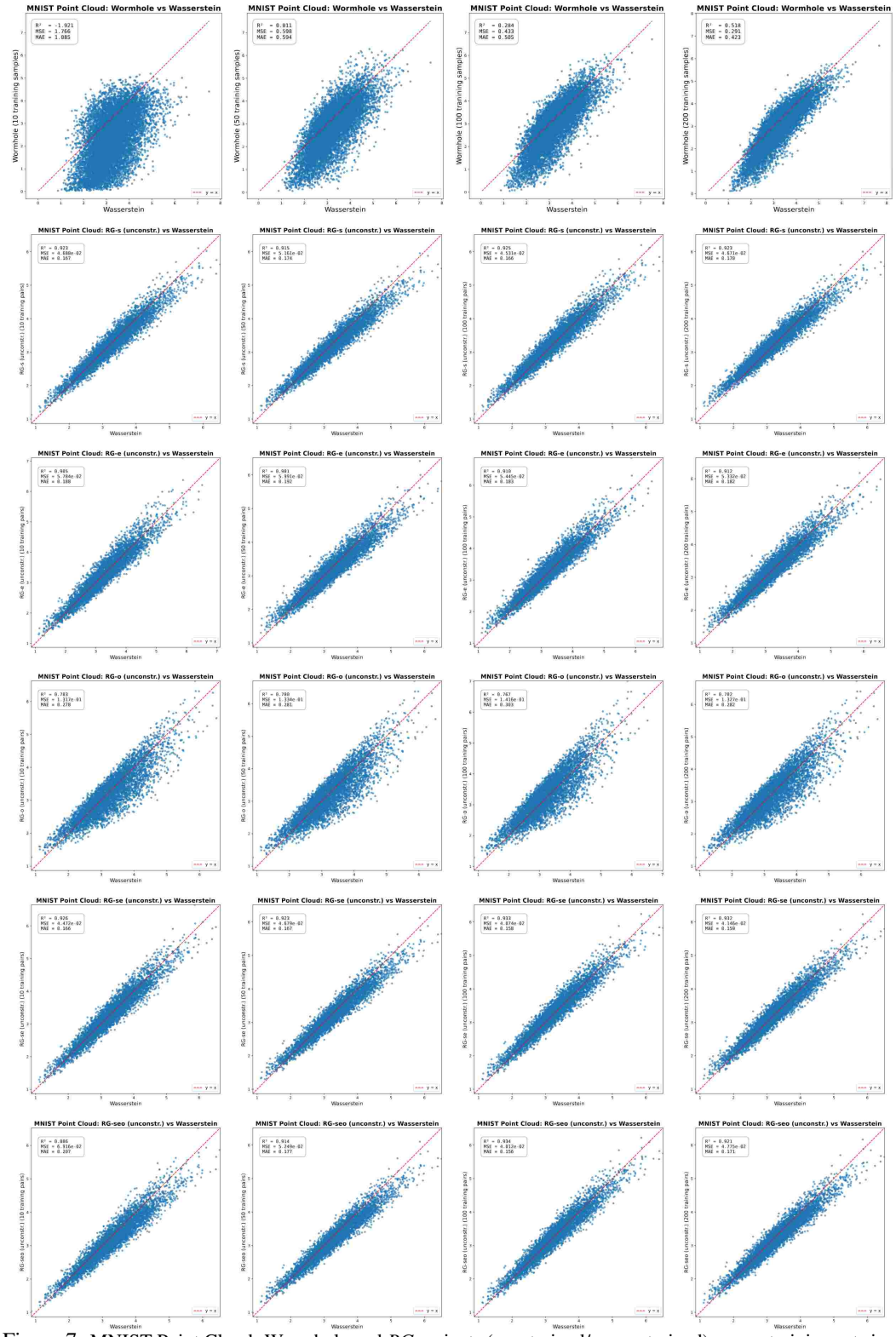

Figure 7: MNIST Point Cloud: Wormhole and *RG* variants (constrained/unconstrained) across training set sizes of 10, 50, 100 and 200.

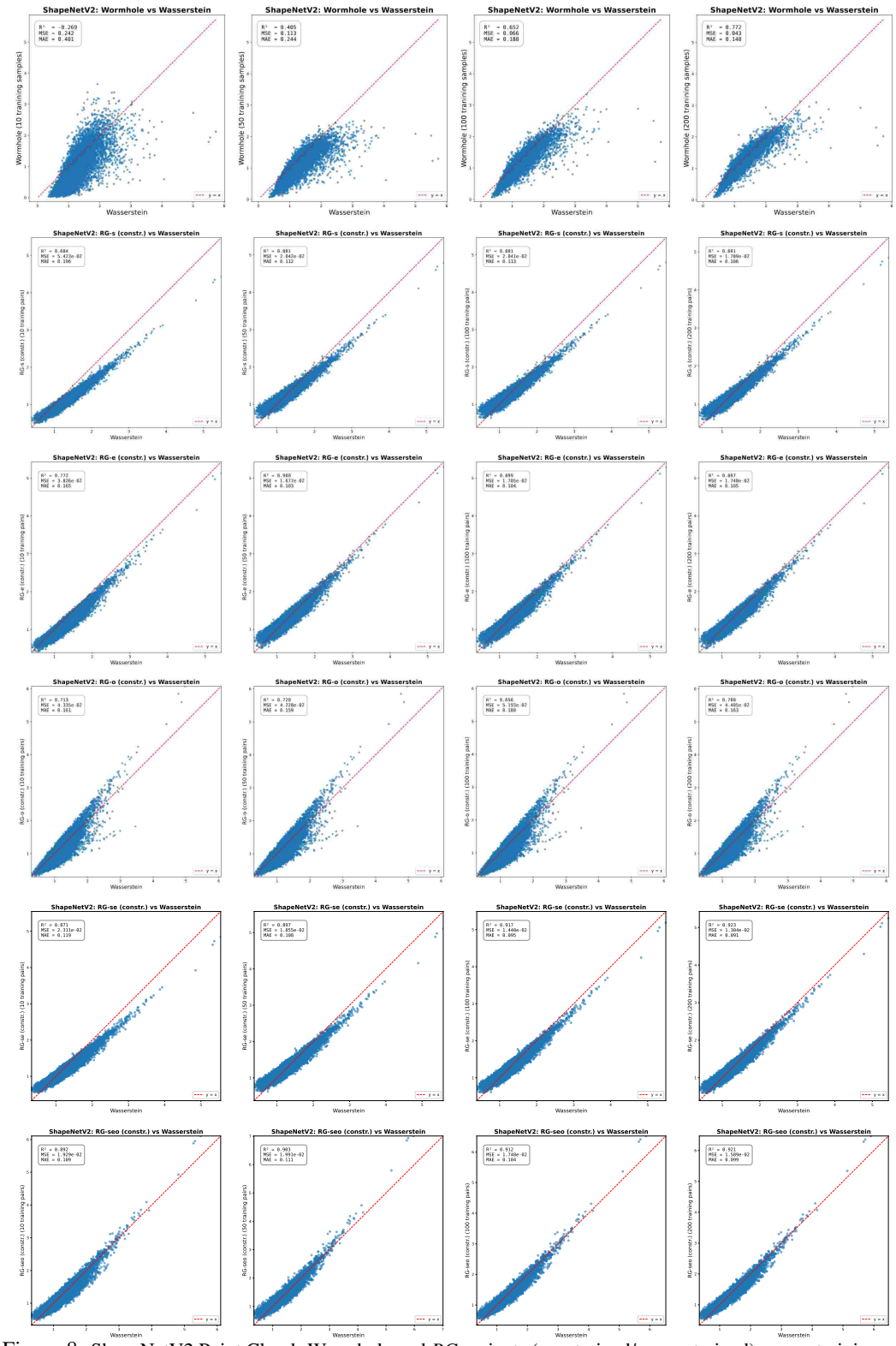

Figure 8: ShapeNetV2 Point Cloud: Wormhole and *RG* variants (constrained/unconstrained) across training set sizes of 10, 50, 100, and 200.

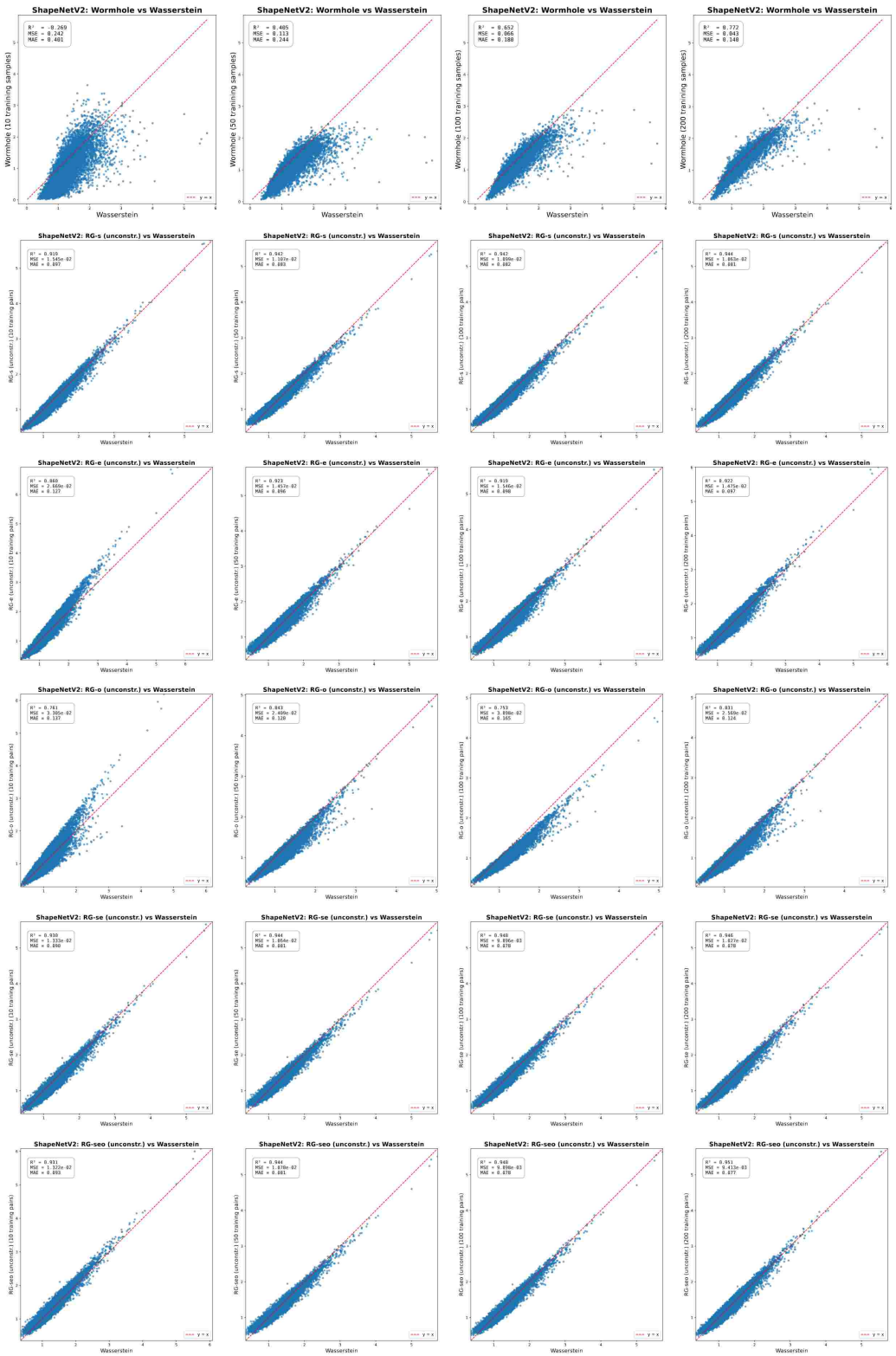

Figure 9: ShapeNetV2 Point Cloud: Wormhole and *RG* variants (constrained/unconstrained) across training set sizes of 10, 50, 100, and 200.

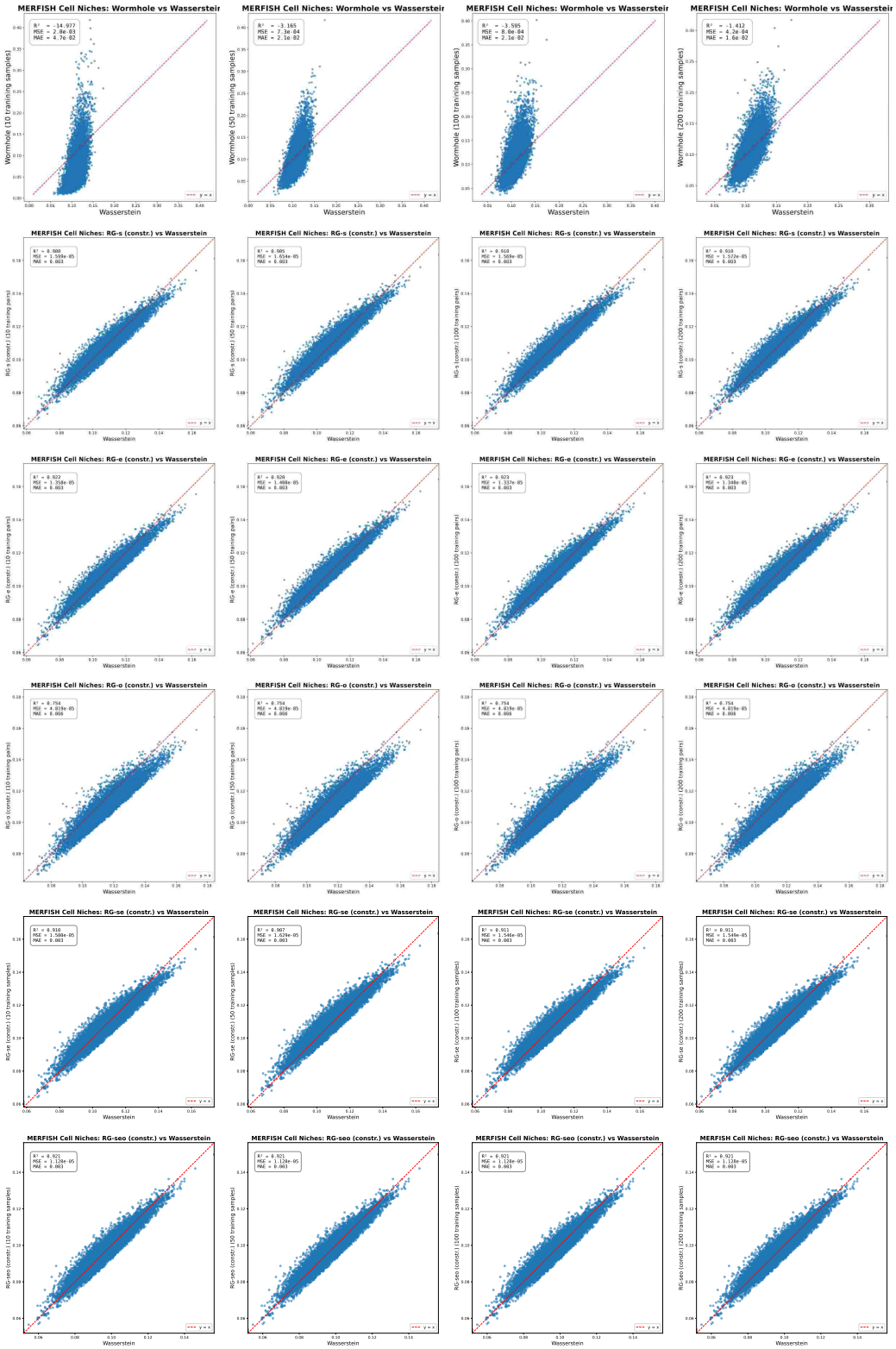

Figure 10: MERFISH Cell Niches: Wormhole and *RG* variants (constrained/unconstrained) across training set sizes of 10, 50, 100, and 200.

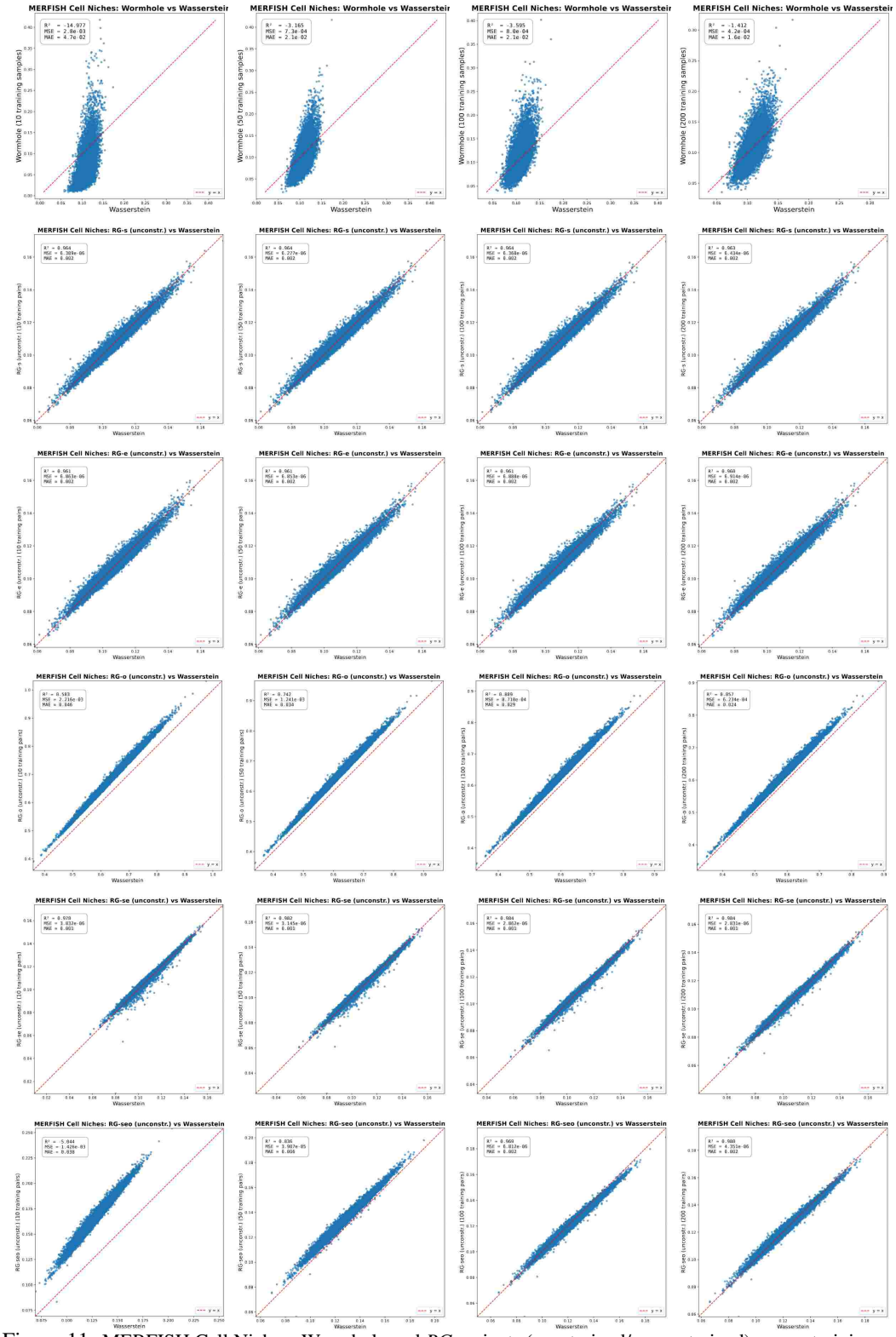

Figure 11: MERFISH Cell Niches: Wormhole and *RG* variants (constrained/unconstrained) across training set sizes of 10, 50, 100, and 200.

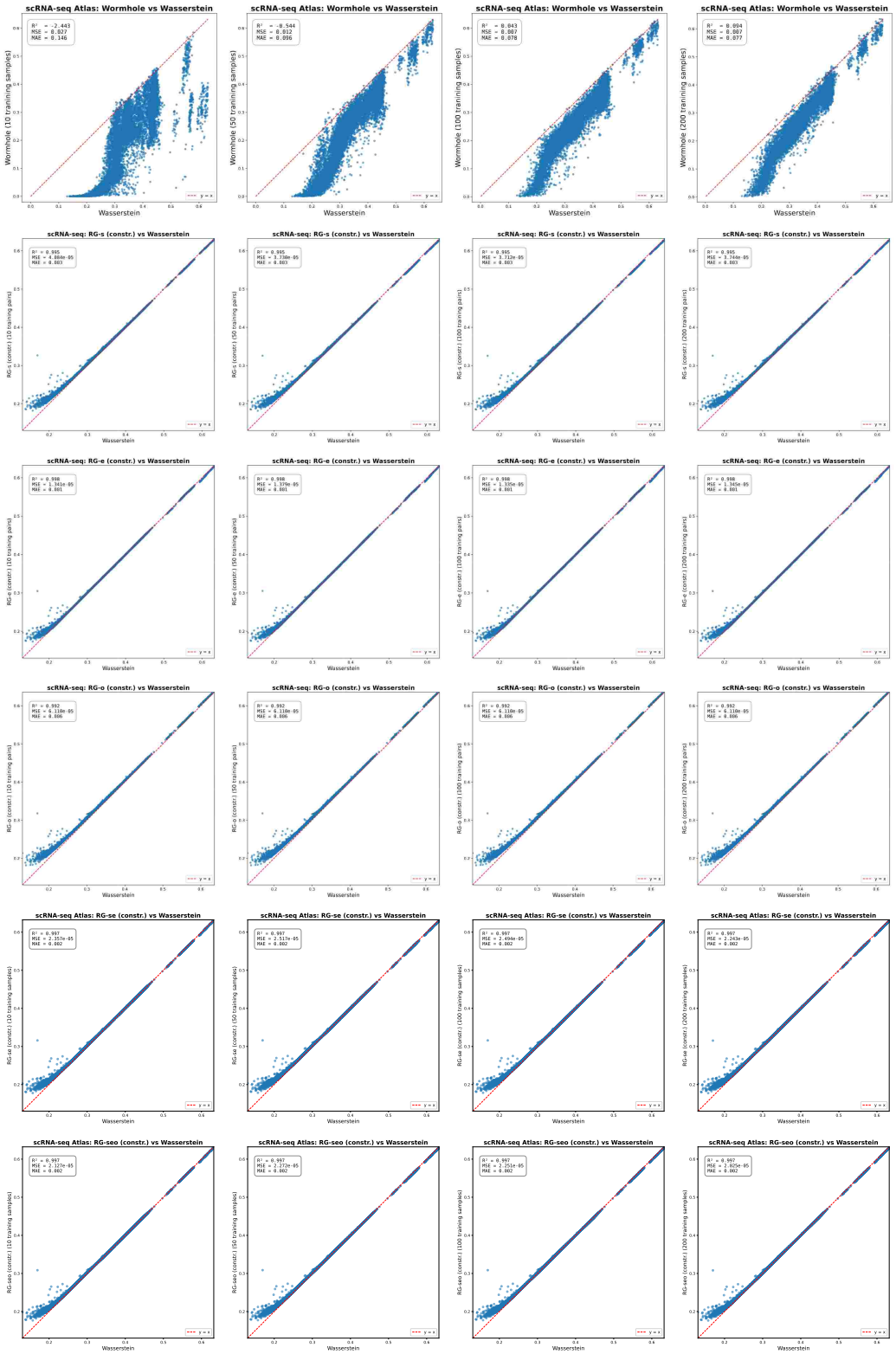

Figure 12: scRNA-seq: Wormhole and *RG* variants (constrained/unconstrained) across training set sizes of 10, 50, 100, and 200.

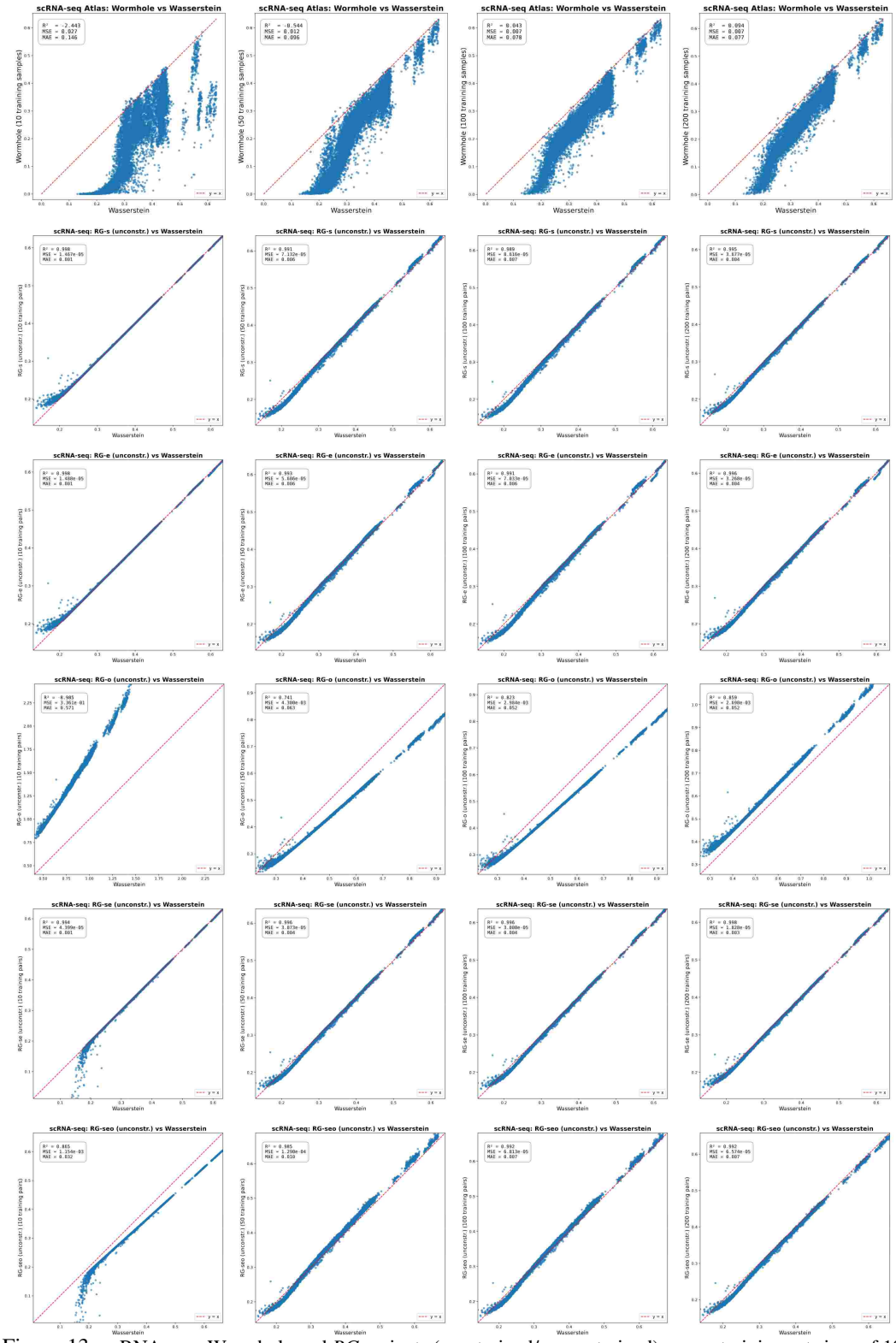

Figure 13: scRNA-seq: Wormhole and *RG* variants (constrained/unconstrained) across training set sizes of 10, 50, 100, and 200.

B.5 RG-WORMHOLE: ACCELERATING WORMHOLE WITH REGRESSION OF WASSERSTEIN

Experimental Settings. We run five experiments to show that *RG-Wormhole* is much faster than Wormhole with similar effectiveness. First, we measure training time by training both models under the same optimizer and schedule, sweeping batch sizes from 4 to 20 and reporting wall-clock time for training sets of 10, 50, 100, and 200 pairs. Second, we assess encoders by computing R^2/MSE/MAE between pairwise distances in the learned embedding space and exact Wasserstein. Third, we evaluate decoders by reporting the Wasserstein loss between each input and its reconstruction. Fourth, we examine barycenters by decoding the mean embedding of each class and visualizing results. Fifth, we study interpolation by decoding linear paths between two embeddings and illustrating trajectories. Across all experiments, hyperparameters match Wormhole; the only change in *RG-Wormhole* is replacing Wasserstein in encoder and decoder losses with the calibrated unconstrained *RG*. We use 10 samples from the training set to estimate RG coefficients. Except for embedding experiment which uses ShapeNetV2 dataset, other experiments use ModelNet40 dataset, same as (Haviv et al., 2024).

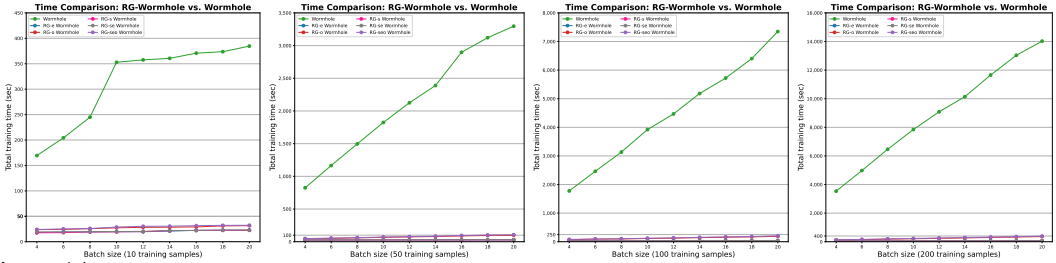

Figure 14: Training time comparison of Wormhole and *RG-Wormhole* methods on point cloud datasets with varying number of training samples.

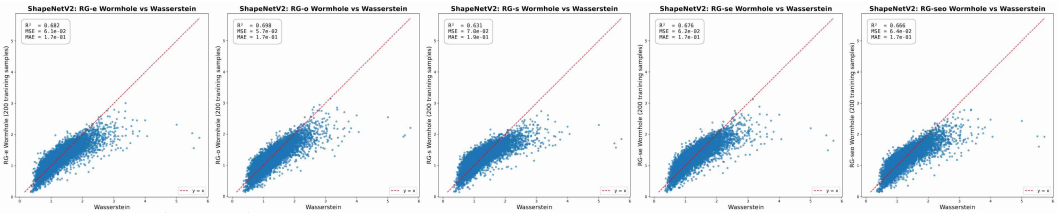

Figure 15: ShapeNetV2: *RG-Wormhole* (constrained model) vs. Wormhole

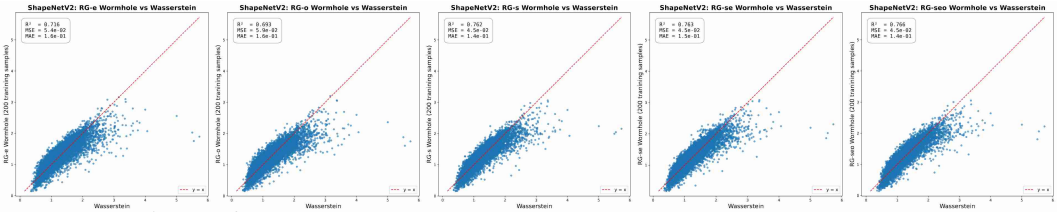

Figure 16: ShapeNetV2: *RG-Wormhole* (unconstrained model) vs. Wormhole

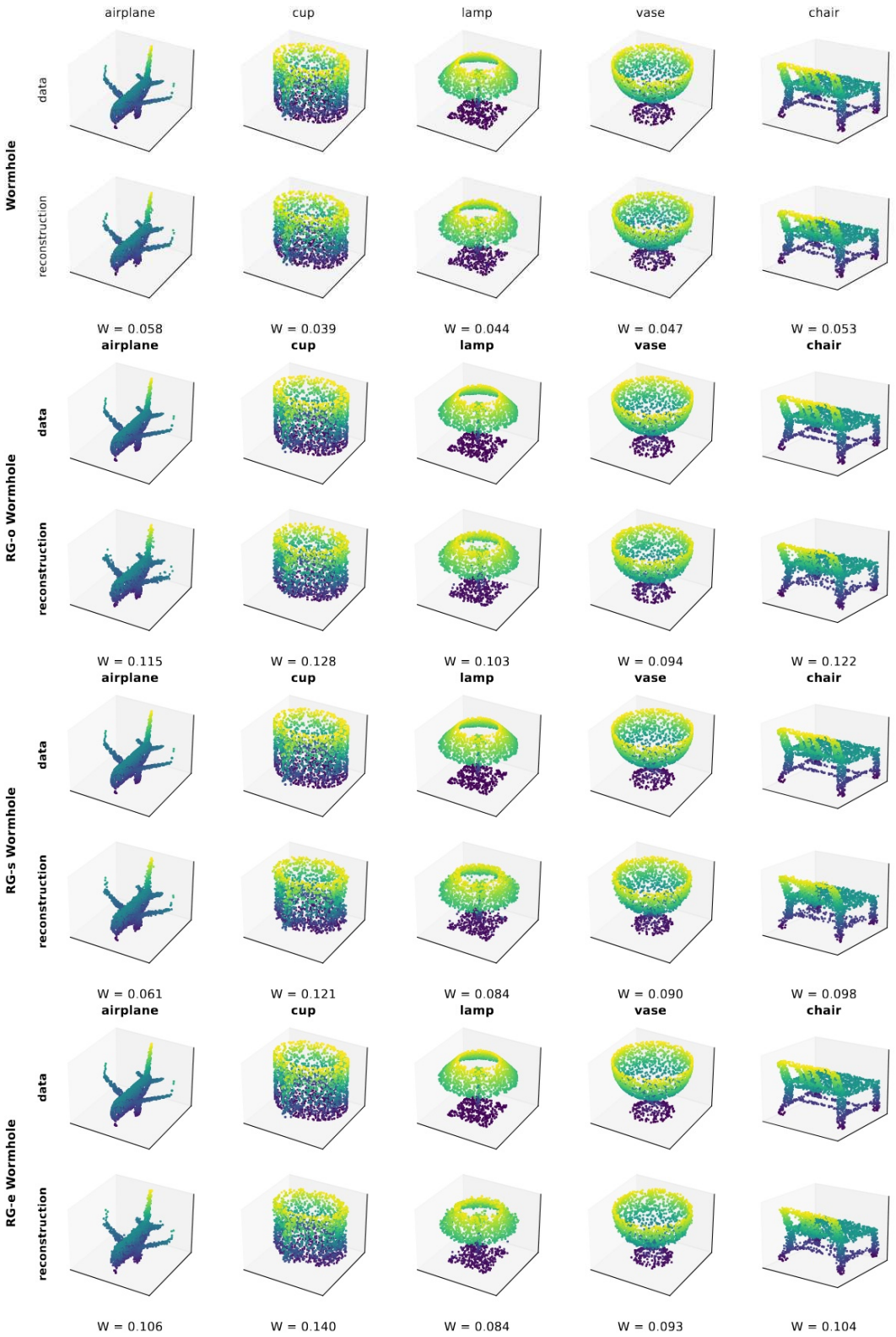

Figure 17: ModelNet40: *RG-Wormhole* vs Wormhole reconstruction experiment

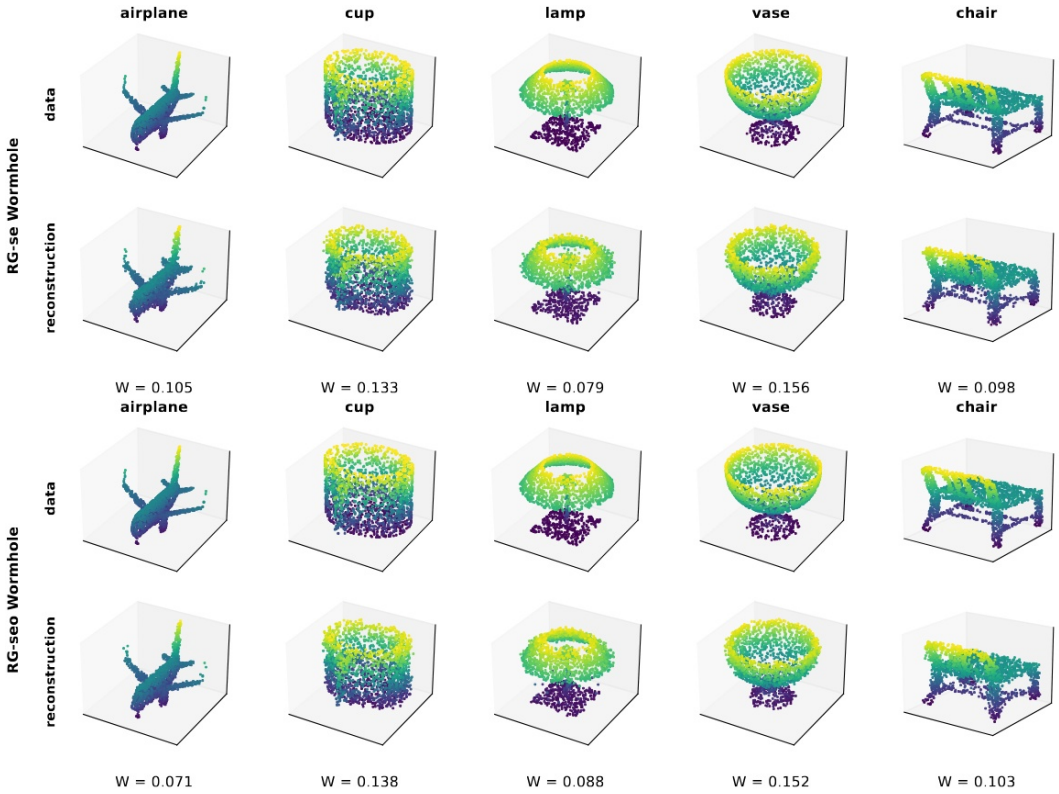

Figure 18: ModelNet40: *RG-Wormhole* reconstruction experiment

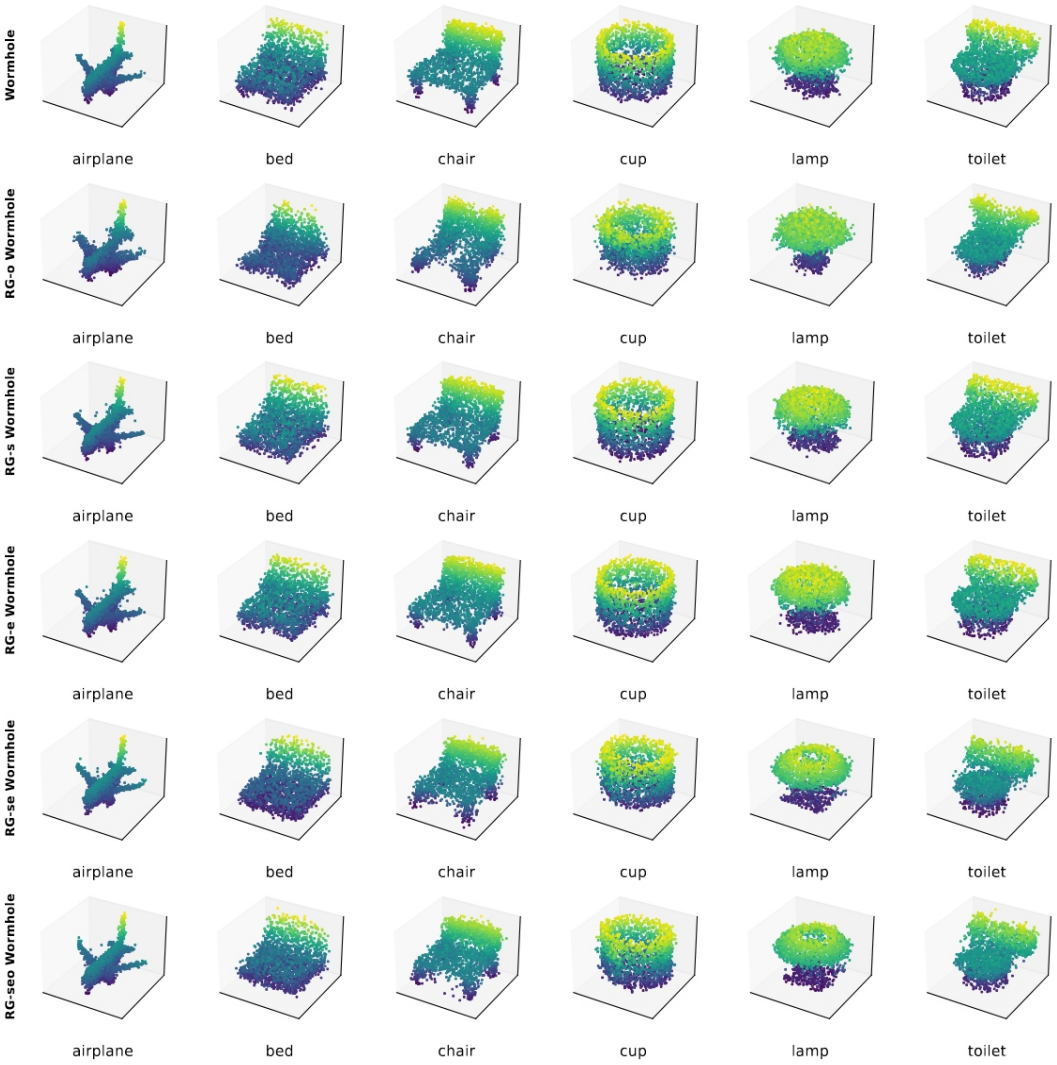

Figure 19: ModelNet40: *RG-Wormhole* barycenter experiment

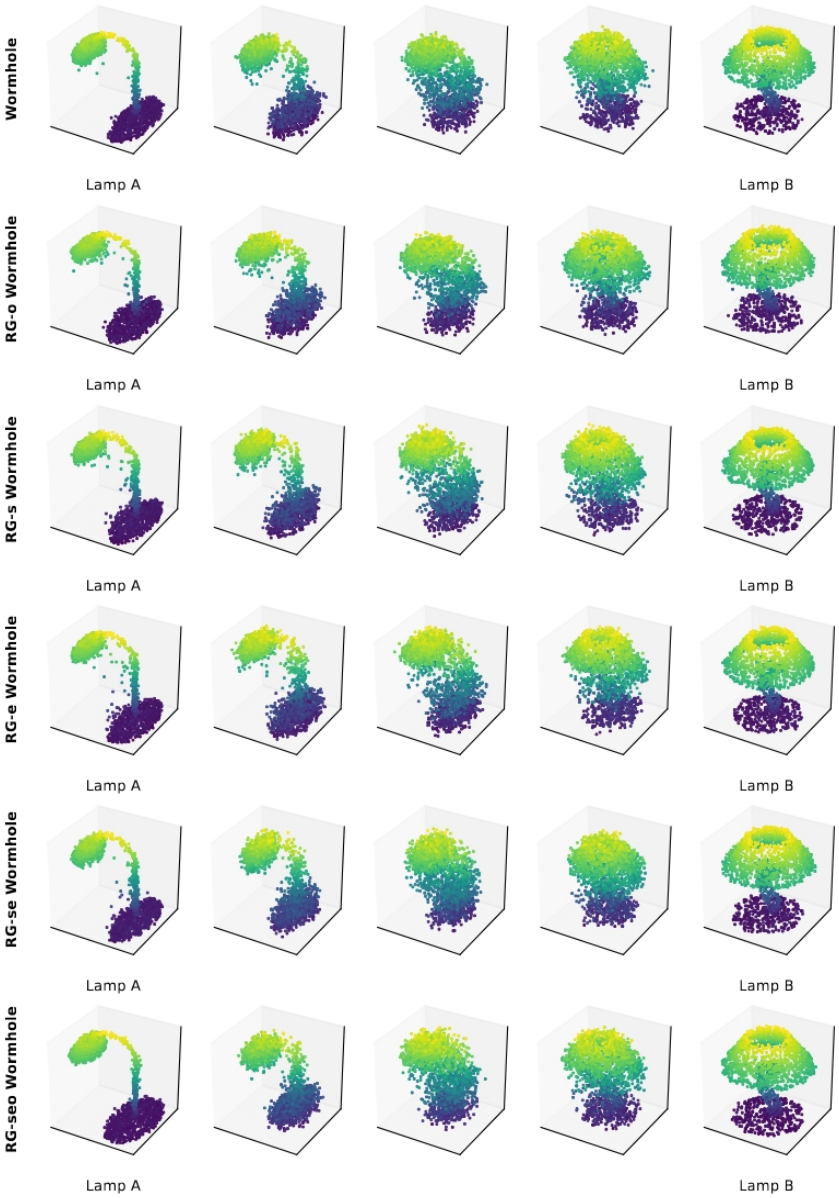

Figure 20: ModelNet40: *RG-Wormhole* barycenter experiment

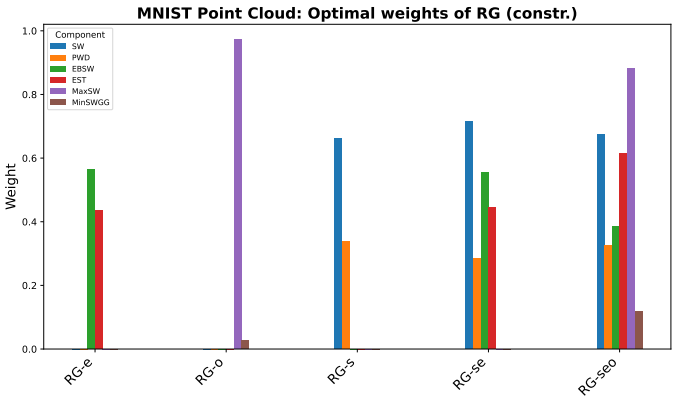

Figure 21: MNIST Point Cloud: Optimal weight of *RG* variants (constrained) across different training samples.

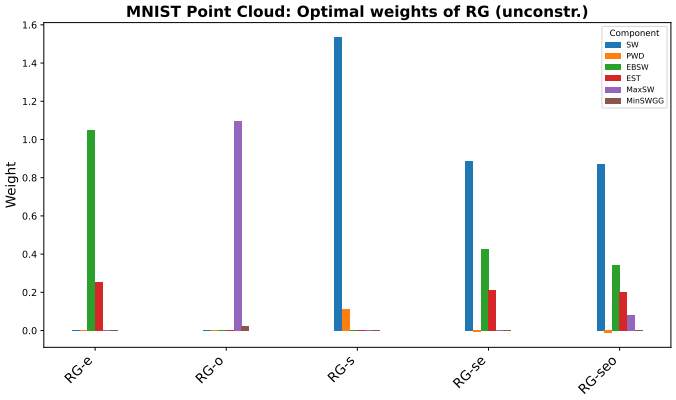

Figure 22: MNIST Point Cloud: Optimal weight of *RG* variants (unconstrained) across different training samples.

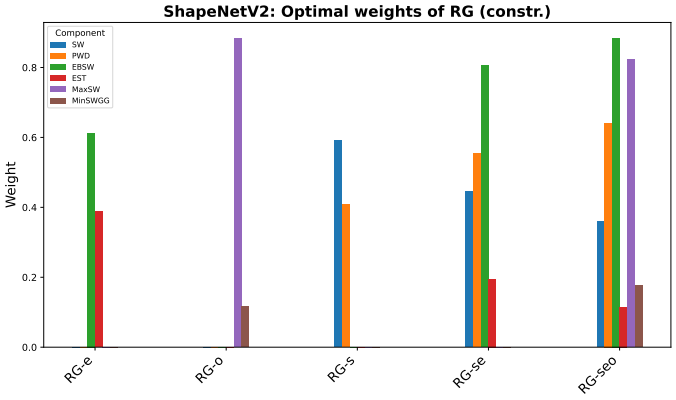

Figure 23: ShapeNetV2: Optimal weight of *RG* variants (constrained) across different training samples.

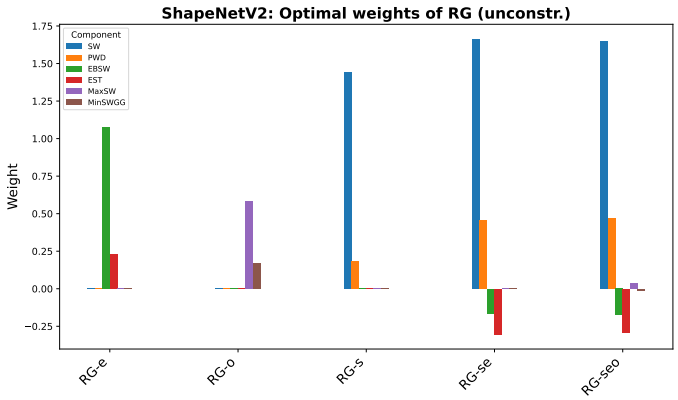

Figure 24: ShapeNetV2: Optimal weight of *RG* variants (unconstrained) across different training samples.

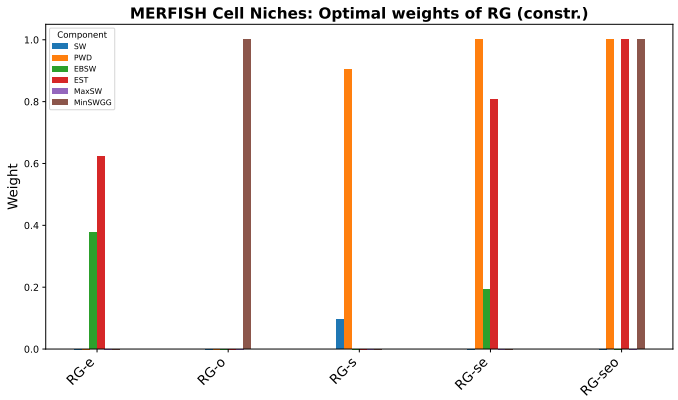

Figure 25: MERFISH Cell Niches: Optimal weight of *RG* variants (constrained) across different training samples.

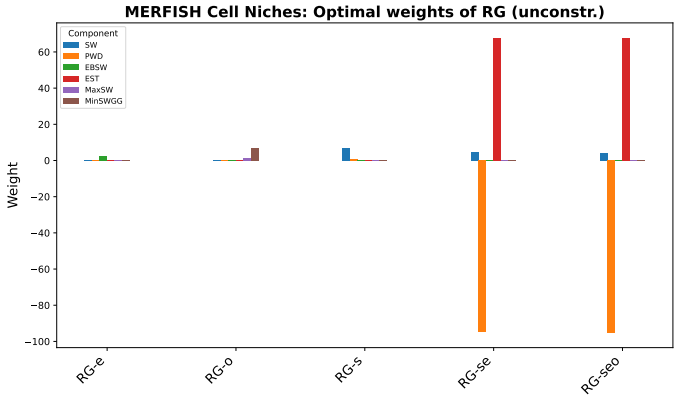

Figure 26: MERFISH Cell Niches: Optimal weight of *RG* variants (unconstrained) across different training samples.

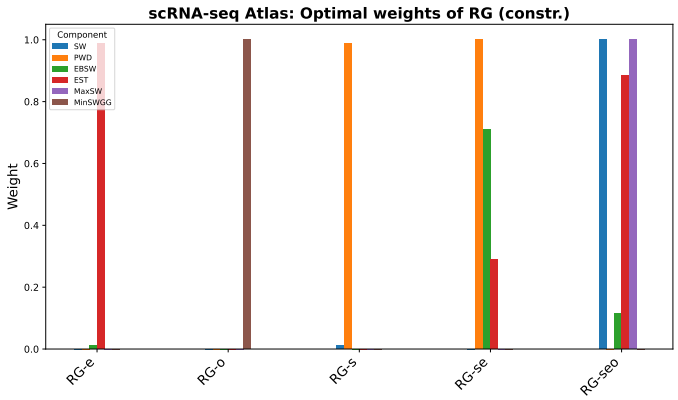

Figure 27: scRNA-seq Atlas: Optimal weight of *RG* variants (constrained) across different training samples.

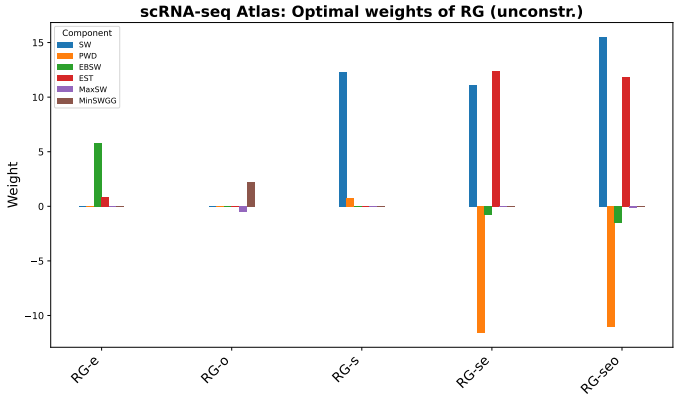

Figure 28: scRNA-seq Atlas: Optimal weight of *RG* variants (unconstrained) across different training samples.

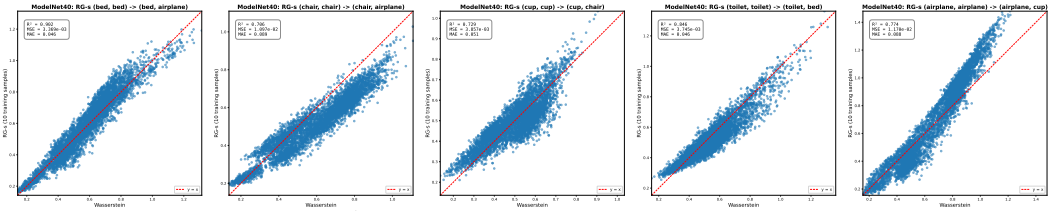

Figure 29: ModelNet40 Intra class

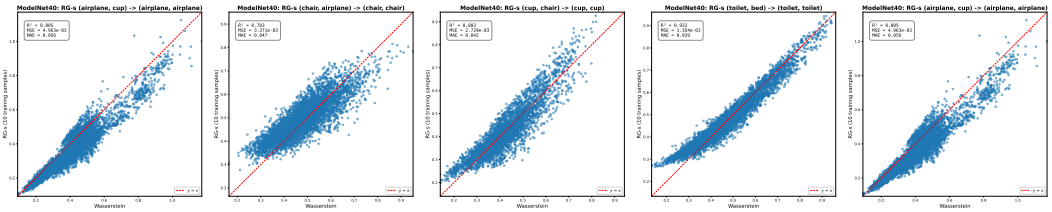

Figure 30: ModelNet40 Inter class