On Deep Generative Models for Approximation and Estimation of Distributions on Manifolds

Biraj Dahal*
School of Mathematics
Georgia Institute of Technology
bdahal6@gatech.edu

Alex Havrilla*
School of Mathematics
Georgia Institute of Technology
ahavrilla3@gatech.edu

Minshuo Chen
Electrical and Computer Engineering
Princeton University
mc0750@princeton.edu

Tuo Zhao
Industrial and Systems Engineering
Georgia Institute of Technology
tourzhao@gatech.edu

Wenjing Liao
School of Mathematics
Georgia Institute of Technology
wliao60@gatech.edu

Abstract

Generative networks have experienced great empirical successes in distribution learning. Many existing experiments have demonstrated that generative networks can generate high-dimensional complex data from a low-dimensional easy-to-sample distribution. However, this phenomenon can not be justified by existing theories. The widely held manifold hypothesis speculates that real-world data sets, such as natural images and signals, exhibit low-dimensional geometric structures. In this paper, we take such low-dimensional data structures into consideration by assuming that data distributions are supported on a low-dimensional manifold. We prove statistical guarantees of generative networks under the Wasserstein-1 loss. We show that the Wasserstein-1 loss converges to zero at a fast rate depending on the intrinsic dimension instead of the ambient data dimension. Our theory leverages the low-dimensional geometric structures in data sets and justifies the practical power of generative networks. We require no smoothness assumptions on the data distribution which is desirable in practice.

1 Introduction

Deep generative models, such as generative adversarial networks (GANs) [Goodfellow et al., 2014, Arjovsky et al., 2017] and variational autoencoder [Kingma and Welling, 2013, Mohamed and Wierstra, 2014], utilize neural networks to generate new samples which follow the same distribution as the training data. They have been successful in many applications including producing photorealistic images, improving astronomical images, and modding video games [Reed et al., 2016, Ledig et al., 2017, Schawinski et al., 2017, Brock et al., 2018, Volz et al., 2018, Radford et al., 2015, Salimans et al., 2016].

To estimate a data distribution Q, generative models solve the following optimization problem

$$\min_{g_\theta \in \mathcal{G}} \texttt{discrepancy}((g_\theta)_\sharp \rho, Q), \tag{1}$$

*These authors contributed equally to this work.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

where ρ is an easy-to-sample distribution, $\mathcal{G}$ is a class of generating functions, `discrepancy` is some distance function between distributions, and $(g_\theta)_\sharp \rho$ denotes the pushforward measure of ρ under g_θ. In particular, when we obtain a sample z from ρ, we let $g_\theta(z)$ be the generated sample, whose distribution follows $(g_\theta)_\sharp \rho$.

There are many choices of the `discrepancy` function in literature among which Wasserstein distance attracts much attention. The so-called Wasserstein generative models [Arjovsky et al., 2017] consider the Wasserstein-1 distance defined as

$$W_1(\mu, \nu) = \sup_{f \in \mathrm{Lip}_1(\mathbb{R}^D)} \mathbb{E}_{X\sim\mu}[f(X)] - \mathbb{E}_{Y\sim\nu}[f(Y)], \tag{2}$$

where μ, ν are two distributions and $\mathrm{Lip}_1(\mathbb{R}^D)$ consists of 1-Lipschitz functions on $\mathbb{R}^D$. The formulation in (2) is known as the Kantorovich-Rubinstein dual form of Wasserstein-1 distance and can be viewed as an integral probability metric [Müller, 1997].

In deep generative models, the function class $\mathcal{G}$ is often parameterized by a deep neural network class $\mathcal{G}_{\mathrm{NN}}$. Functions in $\mathcal{G}_{\mathrm{NN}}$ can be written in the following compositional form

$$g_\theta(x) = W_L \cdot \sigma(W_{L-1} \ldots \sigma(W_1 x + b_1) + \ldots + b_{L-1}) + b_L, \tag{3}$$

where the W_i's and b_i's are weight matrices and intercepts/biases of corresponding dimensions, respectively, and σ is ReLU activation applied entry-wise: $\sigma(a) = \max(a, 0)$. Here $\theta = \{W_i, b_i\}_{i=1}^L$ denotes the set of parameters.

Solving (1) is prohibitive in practice, as we only have access to a finite collection of samples, $X_1, \ldots, X_n \overset{\text{iid}}{\sim} Q$. Replacing Q by its empirical counterpart $Q_n = \frac{1}{n}\sum_{i=1}^n \delta_{X_i}$, we end up with

$$\hat{g}_n = \operatorname*{argmin}_{g_\theta \in \mathcal{G}_{\mathrm{NN}}} W_1((g_\theta)_\sharp \rho, Q_n). \tag{4}$$

Note that (4) is also known as training deep generative models under the Wasserstein loss in existing deep learning literature [Frogner et al., 2015, Genevay et al., 2018]. It has exhibited remarkable ability in learning complex distributions in high dimensions, even though existing theories cannot fully explain such empirical successes. In literature, statistical theories of deep generative models have been studied in Arora et al. [2017], Zhang et al. [2017], Jiang et al. [2018], Bai et al. [2018], Liang [2017, 2018], Uppal et al. [2019], Chen et al. [2020], Lu and Lu [2020], Block et al. [2021], Luise et al. [2020], Schreuder et al. [2021]. Due to the well-known curse of dimensionality, the sample complexity in Liang [2017], Uppal et al. [2019], Chen et al. [2020], Lu and Lu [2020] grows exponentially with respect to underlying the data dimension. For example, the CIFAR-10 dataset consists of 32×32 RGB images. Roughly speaking, to learn this data distribution with accuracy ϵ, the sample size is required to be ϵ^{-D} where $D = 32 \times 32 \times 3 = 3072$ is the data dimension. Setting $\epsilon = 0.1$ requires 10^{3072} samples. However, GANs have been successful with $60,000$ training samples [Goodfellow et al., 2014].

A common belief to explain the aforementioned gap between theory and practice is that practical data sets exhibit low-dimensional intrinsic structures. For example, many image patches are generated from the same pattern by some transformations, such as rotation, translation, and skeleton. Such a generating mechanism induces a small number of intrinsic parameters. It is plausible to model these data as samples near a low dimensional manifold [Tenenbaum et al., 2000, Roweis and Saul, 2000, Peyré, 2009, Coifman et al., 2005].

To justify that deep generative models can adapt to low-dimensional structures in data sets, this paper focuses (from a theoretical perspective) on the following fundamental questions of both distribution approximation and estimation:

Q1: *Can deep generative models approximate a distribution on a low-dimensional manifold by representing it as the pushforward measure of a low-dimensional easy-to-sample distribution?*

Q2: *If the representation in **Q1** can be learned by deep generative models, what is the statistical rate of convergence in terms of the sample size n?*

This paper provides positive answers to these questions. We consider data distributions supported on a d-dimensional compact Riemannian manifold $\mathcal{M}$ isometrically embedded in $\mathbb{R}^D$. The easy-to-sample distribution ρ is uniform on $(0, 1)^{d+1}$. To answer **Q1**, our Theorem 1 proves that deep generative

models are capable of approximating a transportation map which maps the low-dimensional uniform distribution ρ to a large class of data distributions on $\mathcal{M}$. To answer **Q2**, our Theorem 2 shows that the Wasserstein-1 loss in distribution learning converges to zero *at a fast rate depending on the intrinsic dimension* d instead of the data dimension D. In particular we prove that

$$\mathbb{E}W_1((\hat{g}_n)_\sharp \rho, Q) \leqslant Cn^{-\frac{1}{d+\delta}}$$

for all $\delta > 0$ where C is a constant independent of n and D.

Our proof proceeds by constructing an oracle transportation map g^* such that $g^*_\sharp \rho = Q$. This construction crucially relies on a cover of the manifold by geodesic balls, such that the data distribution Q is decomposed as the sum of local distributions supported on these geodesic balls. Each local distribution is then transported onto lower dimensional sets in $\mathbb{R}^d$ from which we can apply optimal transport theory. We then argue that the oracle g^* can be efficiently approximated by deep neural networks.

We make minimal assumptions on the network, only requiring that g_θ belongs to a neural network class (labelled $\mathcal{G}_{\text{NN}}$) with size depending on some accuracy ϵ. Further, we make minimal assumptions on the data distribution Q, only requiring that it admits a density that is upper and lower bounded. Standard technical assumptions are made on the manifold $\mathcal{M}$.

2 Preliminaries

We establish some notation and preliminaries on Riemannian geometry and optimal transport theory before presenting our proof.

Notation. For $x \in \mathbb{R}^d$, $\|x\|$ is the Euclidean norm, unless otherwise specified. $B_X(0, r)$ is the open ball of radius r in the metric space X. If unspecified, we denote $B(0, r) = B_{\mathbb{R}^d}(0, r)$. For a function $f : \mathbb{R}^d \to \mathbb{R}^d$ and $A \subseteq \mathbb{R}^d$, $f^{-1}(A)$ denotes the pre-image of A under f. ∂ denotes the differential operator. For $0 < \alpha \leqslant 1$, we denote by C^α the class of Hölder continuous functions with Hölder index α. $\|\cdot\|_\infty$ denotes the ∞ norm of a function, vector, or matrix (considered as a vector). For any positive integer $N \in \mathbb{N}$, we denote by $[N]$ the set $\{1, 2, \ldots, N\}$.

2.1 Riemannian Geometry

Let $(\mathcal{M}, g)$ be a d-dimensional compact Riemannian manifold isometrically embedded in $\mathbb{R}^D$. Roughly speaking a manifold is a set which is locally Euclidean i.e. there exists a function ϕ continuously mapping a small patch on $\mathcal{M}$ into Euclidean space. This can be formalized with *open sets* and *charts*. At each point $x \in \mathcal{M}$ we have a *tangent space* $T_x\mathcal{M}$ which, for a manifold embedded in $\mathbb{R}^D$, is the d-dimensional plane tangent to the manifold at x. We say $\mathcal{M}$ is Riemannian because it is equipped with a smooth metric $g_x : T_x\mathcal{M} \times T_x\mathcal{M} \to \mathbb{R}$ (where x is a basepoint) which can be thought of as a local inner product. We can define the Riemannian distance $d_\mathcal{M} : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ on $\mathcal{M}$ as

$$d_\mathcal{M}(x, y) = \inf\{L(\gamma) | \gamma \text{ is a } C^1(\mathcal{M}) \text{ curve such that } \gamma(0) = x, \gamma(1) = y\},$$

i.e. the length of the shortest path or *geodesic* connecting x and y. An *isometric embedding* of the d-dimensional $\mathcal{M}$ in $\mathbb{R}^D$ is an embedding that preserves the Riemannian metric of $\mathcal{M}$, including the Riemannian distance. For more rigorous statements, see the classic reference Flaherty and do Carmo [2013].

We next define the exponential map at a point $x \in \mathcal{M}$ going from the tangent space to the manifold.

Definition 1 (Exponential map)**.** *Let $x \in \mathcal{M}$. For all tangent vectors $v \in T_x\mathcal{M}$, there is a unique geodesic γ that starts at x with initial tangent vector v, i.e. $\gamma(0) = x$ and $\gamma'(0) = v$. The exponential map centered at x is given by $\exp_x(v) = \gamma(1)$, for all $v \in T_x\mathcal{M}$.*

The exponential map takes a vector v on the tangent space $T_x\mathcal{M}$ as input. The output, $\exp_x(v)$, is the point on the manifold obtained by travelling along a geodesic curve that starts at x and has initial direction v (see Figure 1 for an example).

It is well known that for all $x \in \mathcal{M}$, there exists a radius δ such that the exponential map restricted to $B_{T_x\mathcal{M}}(0, \delta)$ is a diffeomorphism onto its image, i.e. it is a smooth map with smooth inverse. As

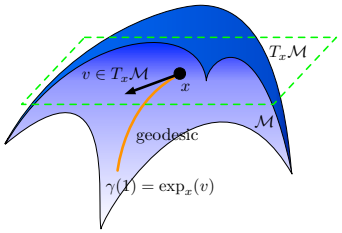


Figure 1: Exponential map on $\mathcal{M}$.

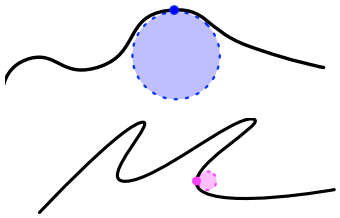


Figure 2: Manifolds with large and small reach.

the sufficiently small δ-ball in the tangent space may vary for each $x \in \mathcal{M}$, we define the injectivity radius of $\mathcal{M}$ as the minimum δ over all $x \in \mathcal{M}$.

Definition 2 (Injectivity radius)**.** *For all $x \in \mathcal{M}$, we define the injectivity radius at x to be $inj_{\mathcal{M}}(x) = \sup\{\delta > 0 | \exp_x : B_{T_x\mathcal{M}}(0, \delta) \subseteq T_x\mathcal{M} \to \mathcal{M}$ is a diffeomorphism$\}$. Then the injectivity radius of $\mathcal{M}$ is defined as*

$$inj(\mathcal{M}) = \inf\{inj_{\mathcal{M}}(x) | x \in \mathcal{M}\}.$$

For any $x \in \mathcal{M}$, the exponential map restricted to a ball of radius $\text{inj}(\mathcal{M})$ in $T_x\mathcal{M}$ is a well-defined diffeomorphism. Within the injectivity radius, the exponential map is a diffeomorphism between the tangent space and a patch of $\mathcal{M}$, with $\exp^{-1}$ denoting the inverse. Controlling a quantity called reach allows us to lower bound the manifold's injectivity radius.

Definition 3 (Reach)**.** *The reach τ of a manifold $\mathcal{M}$ is defined as the quantity (Federer [1959])*

$$\tau = \inf\{r > 0 : \exists x \neq y \in \mathcal{M}, v \in \mathbb{R}^D \text{ such that } r = \|x - v\| = \|y - v\| = \inf_{z \in \mathcal{M}} \|z - v\|\}.$$

Intuitively, if the distance of a point x to $\mathcal{M}$ is smaller than the reach, then there is a unique point in $\mathcal{M}$ that is closest to x. However, if the distance between x and $\mathcal{M}$ is larger than the reach, then there will no longer be a unique closest point to x in $\mathcal{M}$. For example, the reach of a sphere is its radius. A manifold with large and small reach is illustrated in Figure 2. The reach gives us control over the injectivity radius $\text{inj}(\mathcal{M})$; in particular, we know $\text{inj}(\mathcal{M}) \geqslant \pi\tau$ (see Aamari and Levrard [2019] for proof).

2.2 Optimal Transport Theory

Let μ, ν be absolutely continuous measures on the sets $X, Y \subseteq \mathbb{R}^d$ respectively. We say a function $f : X \to Y$ *transports* μ onto ν if $f_\sharp \mu = \nu$. In words, for all measurable sets $A \subseteq Y$ we have

$$\nu(A) = f_\sharp \mu(A) = \mu\left(f^{-1}(A)\right),$$

where $f^{-1}(A)$ is the pre-image of A under f.

Optimal transport studies the problem of transporting source measures μ on X to target measures ν on Y while minimizing a cost $c : X \times Y \to \mathbb{R}_{\geqslant 0}$. However, the results are largely restricted to transport between measures on the same dimensional Euclidean space. In this paper, we will make use of the main theorem in Caffarelli [1992], in the form presented in Villani [2008].

Proposition 1. *Let $c(x, y) = \|x - y\|^2$ in $\mathbb{R}^d \times \mathbb{R}^d$ and let Ω_1, Ω_2 be nonempty, connected, bounded, open subsets of $\mathbb{R}^d$. Let f_1, f_2 be probability densities on Ω_1 and Ω_2 respectively, with f_1, f_2 bounded from above and below. Assume further that Ω_2 is convex. Then there exists a unique optimal transport map $T : \Omega_1 \to \Omega_2$ for the associated probability measures $\mu(dx) = f_1(x)\, dx$ and $\nu(dy) = f_2(y)\, dy$, and the cost c. Furthermore, we have that $T \in C^\alpha(\Omega_1)$ for some $\alpha \in (0, 1)$.*

This proposition allows to produce Hölder transport maps which can be further approximated with neural networks with size depending on a given accuracy.

To connect optimal transport and Riemannian manifolds, we first define the *volume measure* on a manifold $\mathcal{M}$ and establish integration on $\mathcal{M}$.

Definition 4 (Volume measure)**.** *Let $\mathcal{M}$ be a compact d-dimensional Riemannian manifold. We define the volume measure $\mu_{\mathcal{M}}$ on $\mathcal{M}$ as the restriction of the d-dimensional Hausdorff measure $\mathcal{H}^d$.*

A definition for the restriction of the Hausdorff measure can be found in Federer [1959].

We say that the distribution Q has density q if the Radon-Nikodym derivative of Q with respect to $\mu_{\mathcal{M}}$ is q. According to Evans and Gariepy [1992]), for any continuous function $f : \mathcal{M} \to \mathbb{R}$ supported within the image of the ball $B_{T_x\mathcal{M}}(0, \epsilon)$ under the exponential map for $\epsilon < \text{inj}(\mathcal{M})$, we have

$$\int f \, dQ = \int (fq) \, d\mu_{\mathcal{M}} = \int_{B_{T_x\mathcal{M}}(0,\epsilon)} (fq) \circ \exp_x(v) \sqrt{\det g^x_{ij}(v)} \, dv. \tag{5}$$

Here $g^x_{ij}(v) = \langle \partial \exp_x(v)[e_i], \partial \exp_x(v)[e_j] \rangle$ with $(e_1, ..., e_d)$ an orthonormal basis of $T_x\mathcal{M}$.

3 Main Results

We will present our main results in this section, including an approximation theory for a large class of distributions on a Riemannian manifold (Theorem 1), and a statistical estimation theory of deep generative networks for distribution learning (Theorem 2).

We make some regularity assumptions on a manifold $\mathcal{M}$ and assume the target data distribution Q is supported on $\mathcal{M}$. The easy-to-sample distribution ρ is taken to be uniform on $(0, 1)^{d+1}$.

Assumption 1. *$\mathcal{M}$ is a d-dimensional compact Riemannian manifold isometrically embedded in ambient dimension $\mathbb{R}^D$. Via compactness, $\mathcal{M}$ is bounded: there exists $M > 0$ such that $\|x\|_\infty \leqslant M$, $\forall x \in \mathcal{M}$. Further suppose $\mathcal{M}$ has a positive reach $\tau > 0$.*

Assumption 2. *Q is supported on $\mathcal{M}$ and has a density q with respect to the volume measure on $\mathcal{M}$. Further we assume boundedness of q i.e. there exists some constants $c, C > 0$ such that $c \leqslant q \leqslant C$.*

To justify the representation power of feedforward ReLU networks for learning the target distribution Q, we explicitly construct a neural network generator class, such that a neural network function in this generator class can pushfoward ρ to a good approximation of Q.

Consider the following generator class $\mathcal{G}_{\text{NN}}$

$$\begin{aligned}\mathcal{G}_{\text{NN}}(L, p, \kappa) = \{g = [g_1, ..., g_D] : \mathbb{R}^{d+1} \to \mathbb{R}^D | g_j \text{ in form (3) with at most } L \text{ layers} \\ \text{and max width } p, \text{ while } ||W_i||_\infty \leqslant \kappa, ||b_i||_\infty \leqslant \kappa \text{ for all } i \in [L], j \in [D]\},\end{aligned}$$

where $\|\cdot\|_\infty$ is the maximum magnitude in a matrix or vector. The width of a neural network is the largest dimension (i.e. number of rows/columns) among the W_i's and b_i's.

Theorem 1 (Approximation Power of Deep Generative Models)**.** *Suppose $\mathcal{M}$ and Q satisfy Assumptions 1 and 2 respectively. The easy-to-sample distribution ρ is taken to be uniform on $(0, 1)^{d+1}$. Then there exists a constant $0 < \alpha < 1$ (independent of D) such that for any $0 < \epsilon < 1$, there exists a $g_\theta \in \mathcal{G}_{\text{NN}}(L, p, \kappa)$ with parameters*

$$L = O\left(\log\left(\frac{1}{\epsilon}\right)\right), \quad p = O\left(D\epsilon^{-\frac{d}{\alpha}}\right), \quad \kappa = M$$

that satisfies

$$W_1((g_\theta)_\sharp \rho, Q) < \epsilon.$$

Theorem 1 demonstrates the representation power of deep neural networks for distributions Q on $\mathcal{M}$, which answers Question **Q1**. For a given accuracy ϵ, there exists a neural network g_θ which pushes the uniform distribution on $(0, 1)^{d+1}$ forward to a good approximation of Q with accuracy ϵ. The network size is exponential in the intrinsic dimension d.

We next present a statistical estimation theory to answer Question **Q2**.

Theorem 2 (Statistical Guarantees of Deep Wasserstein Learning)**.** *Suppose $\mathcal{M}$ and Q satisfy Assumption 1 and 2 respectively. The easy-to-sample distribution ρ is taken to be uniform on $(0, 1)^{d+1}$. Let n be the number of samples of $X_i \sim Q$. Choose any $\delta > 0$. Set $\epsilon = n^{-\frac{1}{d+\delta}}$ in Theorem 1 so that the network class $\mathcal{G}_{\text{NN}}(L, p, \kappa)$ has parameters*

$$L = O\left(\log\left(n^{\frac{1}{d+\delta}}\right)\right), \quad p = O\left(Dn^{\frac{d}{\alpha(d+\delta)}}\right), \quad \kappa = M.$$

Then the empirical risk minimizer $\hat{g}_n$ given by (4) *has rate*

$$\mathbb{E}W_1((\hat{g}_n)_\sharp\rho, Q) \leqslant Cn^{-\frac{1}{d+\delta}},$$

where C is a constant independent of n and D.

Additionally this result can be easily extended to the noisy case. Suppose we are given n noisy i.i.d. samples $\hat{X}_1, ..., \hat{X}_n$ of the form $\hat{X}_i = X_i + \xi_i$, for $X_i \overset{\text{iid}}{\sim} Q$ and ξ_i distributed according to some noise distribution. The optimization in (4) is performed with the noisy empirical distribution $\hat{Q}_n = \frac{1}{n}\sum_{i=1}^n \delta_{\hat{X}_i}$. Then the minimizer $\hat{g}_n$ satisfies

$$\mathbb{E}W_1((\hat{g}_n)_\sharp\rho, Q) \leqslant Cn^{-\frac{1}{d+\delta}} + 2\sqrt{V_\xi},$$

where $V_\xi = \mathbb{E}\|\xi\|_2^2$ is the variance of the noise distribution.

Comparison to Related Works. To justify the practical power of generative networks, low-dimensional data structures are considered in Luise et al. [2020], Schreuder et al. [2021], Block et al. [2021], Chae et al. [2021]. These works consider the generative models in (1). They assume that the high-dimensional data are parametrized by low-dimensional latent parameters. Such assumptions correspond to the manifold model where the manifold is globally homeomorphic to Euclidean space, i.e. the manifold has a single chart.

In Luise et al. [2020], the generative models are assumed to be continuously differentiable up to order s. By jointly training of the generator and the latent distributions, they proved that the Sinkhorn divergence between the generated distribution and data distribution converges, depending on data intrinsic dimension. Chae et al. [2021] and Schreuder et al. [2021] assume the special case where the manifold has a single chart. More recently, Block et al. [2021] proposed to estimate the intrinsic dimension of data using the Hölder IPM between some empirical distributions of data. This theory is based on the statistical convergence of the empirical distribution to the data distribution. As an application to GANs, [Block et al., 2021, Theorem 23] gives the statistical error while the approximation error is not studied. In these works, the single chart assumption is very strong while a general manifold can have multiple charts.

Recently, Yang et al. [2022], Huang et al. [2022] showed that GANs can approximate any data distribution (in any dimension) by transforming an absolutely continuous 1D distribution. The analysis in Yang et al. [2022], Huang et al. [2022] can be applied to the general manifold model. Their approach requires the GAN to memorize the empirical data distribution using ReLU networks. Thus it is not clear how the designed generator is capable of generating new samples different from the training data.

In contrast, we explicitly construct an oracle transport map which transforms the low-dimensional easy-to-sample distribution to the data distribution. Our work provides insights about how distributions on a manifold can be approximated by a neural network pushforward of a low-dimensional easy-to-sample distribution without exactly memorizing the data. In comparison, the single-chart assumption in earlier works assumes that an oracle transport naturally exists. Our work is novel in the construction of the oracle transport for a general manifold with multiple charts, and the approximation theory by deep neural networks.

4 Proof of Main Results

4.1 Proof of Approximation Theory in Theorem 1

To prove Theorem 1, we explicitly construct an oracle transport g^* pushing ρ onto Q, i.e. $g^*_\sharp\rho = Q$. Further this oracle will be piecewise α-Hölder continuous for some $\alpha \in (0, 1)$.

Lemma 1. *Suppose $\mathcal{M}$ and Q satisfy Assumption 1 and 2 respectively. The easy-to-sample distribution ρ is taken to be uniform on $(0,1)^{d+1}$. Then there exists a function $g^* : (0,1)^{d+1} \to \mathcal{M}$ such that $Q = g^*_\sharp\rho$ where*

$$g^*(x) = \textstyle\sum_{j=1}^J \mathbb{1}_{(\pi_{j-1},\pi_j)}(x_1) g_j^*(x_{2:d+1}) \tag{6}$$

for some α-Hölder ($0 < \alpha < 1$) continuous functions $g_1^*, \dots, g_J^*$ and some constants $0 = \pi_0 < \pi_1 < \cdots < \pi_J = 1$.

Proof. We construct a transport map $g^* : (0,1)^{d+1} \to \mathcal{M}$ that can be approximated by neural networks. First, we decompose the manifold into overlapping geodesic balls. Next, we pull these local distributions on these balls back to tangent space, which produces d-dimensional tangent distributions. Then, we apply optimal theory on these tangent distributions to produce maps between the source distributions on $(0,1)^d$ to the appropriate local (geodesic ball) distributions on the manifold. Finally, we glue together these local maps with indicators functions and a uniform random sample from $(0,1)$. We proceed with the first step of decomposing the manifold.

Step 1: Overlapping ball decomposition. Recall that $\mathcal{M}$ is a compact manifold with reach $\tau > 0$. Then the injectivity radius of $\mathcal{M}$ is greater than or equal to $\pi\tau$ (Aamari et al. [2019]). Set $r = \frac{\pi\tau}{2}$. For each $c \in \mathcal{M}$, define an open set $U_c = \exp_c(B_{T_c\mathcal{M}}(0,r)) \subseteq \mathcal{M}$. Since the collection $\{U_c : x \in \mathcal{M}\}$ forms an open cover of $\mathcal{M}$ (in $\mathbb{R}^D$), by the compactness of $\mathcal{M}$ we can extract a finite subcover which we denote as $\{U_{c_j}\}_{j=1}^J$. For convenience, we will write $U_j = U_{c_j}$.

Step 2: Defining local lower-dimensional distributions. On each U_j, we define a local distribution Q_j with density q_j via

$$q_j(x) = \frac{q(x)}{Q(U_j)}\mathbb{1}_{U_j}(x).$$

Set $K(x) = \sum_{j=1}^J \mathbb{1}_{U_j}(x)$ as the number of balls U_j containing x. Note $1 \leqslant K(x) \leqslant J$ for all $x \in \mathcal{M}$. Now define the distribution $\overline{Q}_j$ with density $\overline{q}_j$ given by

$$\overline{q}_j(x) = \frac{\frac{1}{K(x)} q_j(x)\mathbb{1}_{U_j}(x)}{\int_{U_j} \frac{1}{K(x)} q_j(x) d\mathcal{H}}.$$

Write $K_j = \int_{U_j} \frac{1}{K(x)} q_j(x) d\mathcal{H}$ as the normalizing constant. Define $\tilde{q}_j(v) = (\overline{q}_j \circ \exp_{c_j})(v)\sqrt{\det g_{kl}^{c_j}(v)}$ where $g_{kl}^{c_j}$ is the Riemannian at c_j. This quantity can be thought of as the Jacobian of the exponential map, denoted by $|J_{\exp_{c_j}}(v)|$ in the following step. Then $\tilde{q}_j$ is a density on $\tilde{U}_j = \exp_{c_j}^{-1}(U_j)$, which is a ball of radius $\frac{\pi\tau}{2}$ since

$$1 = \int_{U_j} \overline{q}_j(x) d\mathcal{H} = \int_{\tilde{U}_j} \sqrt{\det g_{kl}^{c_j}(v)}\overline{q}_j(\exp_{c_j}(v)) dv = \int_{\tilde{U}_j} \tilde{q}_j(v) dv$$

Let $\tilde{Q}_j$ be the distribution in $\mathbb{R}^d$ with density $\tilde{q}_j$. By construction, we can write

$$\overline{Q}_j = (\exp_{c_j})_\sharp \tilde{Q}_j. \tag{7}$$

Step 3: Constructing the local transport. We have that $\exp_{c_j}^{-1}$ is bi-Lipschitz on U_j and hence its Jacobian is upper bounded. Since $|J_{\exp_{c_j}}(v)| = \frac{1}{|J_{\exp_{c_j}^{-1}(x)}|}$, we know that $|J_{\exp_{c_j}}|$ lower bounded. Since q_j is lower bounded (away from 0), this means $\tilde{q}_j$ is also lower bounded. Now the distribution $\tilde{\rho}_j$ supported on $\tilde{U}_j = B(0, \frac{\tau\pi}{2})$ fulfills the requirements for our optimal transport result: (1) Its density $\tilde{p}_j$ is lower and upper bounded; (2) The support $B(0, \frac{\tau\pi}{2})$ is convex. Taking our cost to be $c(x,y) = \frac{1}{2}\|x-y\|^2$ (i.e. squared Euclidean distance), via Proposition 1 we can find an optimal transport map T_j such that

$$(T_j)_\sharp \rho_d = \tilde{Q}_j \tag{8}$$

where ρ_d is uniformly distributed on $(0,1)^d$. Furthermore, $T_j \in C^{\alpha_j}$ for some $\alpha_j \in (0,1)$. Then we can construct a local transport onto U_j via

$$g_j^* = \exp_{c_j} \circ T_j \tag{9}$$

which pushes ρ_d forward to $\overline{Q}_j$. Since g_j^* is a composition of a Lipschitz map with an α_j Hölder continuous maps, it is hence α_j Hölder continuous.

Step 4: Assembling the global transport. It remains to patch together the local distributions $\overline{Q}_j$ to form Q. Define $\eta_j = K_j Q(U_j)$. Notice

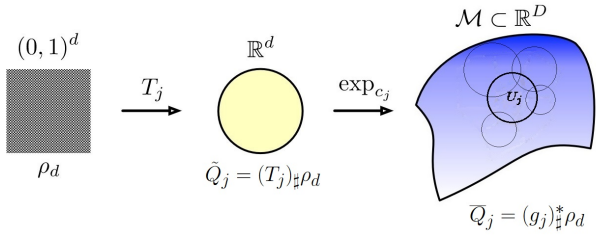


Figure 3: Local transport g_j^* in (9) mapping ρ_d on $(0,1)^d$ to a local distribution $\overline{Q}_j$ supported on U_j.

$$\sum_{j=1}^{J} \eta_j \overline{q}_j(x) = \sum_{j=1}^{J} K_j Q(U_j) \frac{\frac{1}{K(x)} q_j(x) \mathbb{1}_{U_j}(x)}{K_j} = \sum_{j=1}^{J} Q(U_j) \frac{\frac{1}{K(x)} q(x) \mathbb{1}_{U_j}(x)}{Q(U_j)}$$
$$= \sum_{j=1}^{J} \frac{1}{K(x)} q(x) \mathbb{1}_{U_j}(x) = q(x) \frac{1}{K(x)} \sum_{j=1}^{J} \mathbb{1}_{U_j}(x) = q(x).$$

Hence it must be that $\sum_{j=1}^{J} \eta_j = 1$. Set $\alpha = \min_{j \in [J]} \alpha_j$. We can now define the oracle g^*. Let $x \in (0,1)^{d+1}$. Write

$$g^*(x) = \textstyle\sum_{j=1}^{J} \mathbb{1}_{(\pi_{j-1}, \pi_j)}(x_1) g_j^*(x_{2:d+1}), \tag{10}$$

where x_1 is the first coordinate and $x_{2:d+1}$ are the remaining coordinates with $\pi_j = \sum_{i=1}^{j-1} \eta_i$. Let $Z \sim \rho$. Then $g(Z) \sim Q$. We see this as follows. For $A \subseteq \mathcal{M}$ we can compute

$$\mathbb{P}(g^*(Z) \in A) = \sum_{j=1}^{J} \mathbb{P}(\pi_{j-1} < Z_1 < \pi_j) \mathbb{P}(g_j^*(Z_{2:d+1}) \in A \cap U_j) = \sum_{j=1}^{J} \eta_j \overline{Q}_j(A \cap U_j)$$
$$= \sum_{j=1}^{J} \eta_j \int_A \overline{q}_j(x) d\mathcal{H} = \int_A \sum_{j=1}^{J} \eta_j \overline{q}_j(x) d\mathcal{H} = \int_A q(x) d\mathcal{H} = Q(A)$$

which completes the proof. □

We have found an oracle g^* which is piecewise Hölder continuous such that $g^*_\sharp \rho = Q$. We can design a neural network g_θ to approximate this oracle g^*. Now in order to minimize $W_1((g_\theta)_\sharp \rho, Q) = W_1((g_\theta)_\sharp \rho, g^*_\sharp \rho)$, we show it suffices to have g_θ approximate g^* in $L^1(\rho)$.

Lemma 2. *Let μ be an absolutely continuous probability distribution on a set $Z \subseteq \mathbb{R}^d$, and let $f, g : Z \to \mathbb{R}^m$ be transport maps. Then*

$$W_1(f_\sharp \mu, g_\sharp \mu) \leqslant C \|f - g\|_{L^1(\mu)}$$

for some $C > 0$.

The proof can be found in Section B.1 in the appendix. We now prove Theorem 1.

Proof of Theorem 1. By Lemma 1, there exists a transformation $g^*(x) = \sum_{j=1}^{J} \mathbb{1}_{(\pi_{j-1}, \pi_j)}(x_1) g_j^*(x_{2:d+1})$ such that $g^*_\sharp \rho = Q$. By Lemma 2, it suffices to approximate g^* with a neural network $g_\theta \in \mathcal{G}_{\text{NN}}(L, p, \kappa)$ in L^1 norm, with a given accuracy $\epsilon > 0$. Let $(g^*)^{(i)}$ denote the ith component of the vector valued function g^*. Then it suffices to approximate

$$(g^*)^{(i)}(x) = \sum_{j=1}^{J} \mathbb{1}_{(\pi_{j-1}, \pi_j)}(x_1) (g_j^*)^{(i)}(x_{2:d+1})$$

for each $1 \leqslant i \leqslant D$, where $(g_j^*)^{(i)}$ denotes the ith component of the function g_j^*. We construct the approximation of $(g^*)^{(i)}$ by the function

$$(g_\theta)^{(i)}(x) = \sum_{j=1}^{J} \tilde{\times}^{\delta_2} \left(\tilde{\mathbb{1}}^{\delta_1}_{(\pi_{j-1},\pi_j)}(x_1), (g_{j,\theta}^{\delta_3})^{(i)}(x_{2:d+1}) \right), \tag{11}$$

where $\tilde{\times}^{\delta_2}$ is a ReLU network approximation to the multiplication operation with δ_2 accuracy, $\tilde{\mathbb{1}}^{\delta_1}_{(\pi_{j-1},\pi_j)}$ is a ReLU network approximation to the indicator function with δ_1 accuracy, and $(g_{j,\theta}^{\delta_3})^{(i)}$ is a ReLU network approximation to $(g_j^*)^{(i)}$ with δ_3 accuracy. We construct these using the approximation theory outlined in Appendix A.

First, we obtain $\tilde{\mathbb{1}}^{\delta_1}_{(\pi_{j-1},\pi_j)}$ via an application of Lemma 9. Next, we obtain $\tilde{\times}^{\delta_2}$ from an application of Lemma 7. Finally, we discuss $g_{j,\theta}^{\delta_3}$. Let $j \in [J]$. To approximate the Hölder function g_j^*, we use the following Lemma 3 that is proved in Appendix A. Similar approximation results can be found in Shen et al. [2022] and Ohn and Kim [2019] as well. In Lemma 3, our approximation error is in L^1 norm and all weight parameters are upper bounded by a constant. In comparison, the error in Ohn and Kim [2019] is in L^∞ norm and the weight parameter increases as ϵ decreases.

Lemma 3. *Fix $M \geqslant 2$. Suppose $f \in C^\alpha([0,1]^d)$, $\alpha \in (0,1]$, with $\|f\|_{L^\infty} < M$. Let $0 < \epsilon < 1$. Then there exists a function Φ implementable by a ReLU network such that*

$$\|f - \Phi\|_{L^1} < \epsilon.$$

The ReLU network has depth at most $c_1 \log\left(\frac{1}{\epsilon}\right)$, width at most $c_2 \epsilon^{-\frac{d}{\alpha}}$, and weights bounded by M (where c_1 and c_2 are constants independent of ϵ).

We can apply Lemma 3 to $(g_j^*)^{(i)}$ for all $1 \leqslant j \leqslant J$ and $1 \leqslant i \leqslant D$, since they are all elements of $C^\alpha(0,1)^d$ and elements of $C^\alpha(0,1)^d$ can be extended to $C^\alpha[0,1]^d$. Thus there exists a neural network $(g_{j,\theta}^{\delta_3})^{(i)} \in \mathcal{G}_{\text{NN}}(L,p,\kappa)$ with parameters given as above such that

$$\|(g_j^*)^{(i)} - (g_{j,\theta}^{\delta_3})^{(i)}\|_{L^1} < \delta_3.$$

The goal is now to show the L^1 distance between g_θ (as defined in (11)) and g^* is small. We have

$$\|g^* - g_\theta\|_{L^1} \leqslant DJ(M\delta_1 + \delta_2 + \delta_3)$$

where δ_1 is the neural network approximation error of indicator functions in L^1, δ_2 is the approximation error of multiplication, and δ_3 is the approximation error of our α-Hölder local transport maps. We carefully argue in Appendix A.4 that each of these components can be approximated with the appropriately sized network. To complete the proof we conclude g_θ can be exactly represented by a neural network in $\mathcal{G}_{\text{NN}}(L,p,\kappa)$ with parameters

$$L = O\left(\log\left(\frac{1}{\epsilon}\right)\right), \quad p = O\left(D\epsilon^{-\frac{d}{\alpha}}\right), \quad \kappa = M.$$

□

4.2 Proof of Statistical Estimation Theory in Theorem 2

The proof of Theorem 2 is facilitated by the common bias-variance inequality, presented here as a lemma.

Lemma 4. *Under the same assumptions of Theorem 2, we have*

$$\mathbb{E}W_1((\hat{g}_n)_\sharp \rho, Q) \leqslant \inf_{g_\theta \in \mathcal{G}_{\text{NN}}} W_1((g_\theta)_\sharp \rho, Q) + 2\mathbb{E}W_1(Q_n, Q) \tag{12}$$

where Q_n is the empirical distribution.

Proof. We compute recalling the definition of $\hat{g}_n$ as the empirical risk minimizer.

$$\begin{aligned}\mathbb{E}W_1((\hat{g}_n)_\sharp\rho, Q) &\leqslant \mathbb{E}W_1((\hat{g}_n)_\sharp\rho, Q_n) + \mathbb{E}W_1(Q_n, Q)\\ &= \mathbb{E}\inf_{g_\theta\in\mathcal{G}_{\text{NN}}} W_1((g_\theta)_\sharp\rho, Q_n) + \mathbb{E}W_1(Q_n, Q)\\ &\leqslant \mathbb{E}\inf_{g_\theta\in\mathcal{G}_{\text{NN}}} W_1((g_\theta)_\sharp\rho, Q) + 2\mathbb{E}W_1(Q_n, Q)\end{aligned}$$

where we recall $W_1((\hat{g}_n)_\sharp\rho, Q_n) = \inf_{g_\theta\in\mathcal{G}_{\text{NN}}} W_1((g_\theta)_\sharp\rho, Q_n)$ from (4).

□

The bias term can be controlled via Theorem 1. To control convergence of the empirical distribution Q_n to Q we leverage the existing theory [Weed and Bach, 2019] to obtain the following lemma.

Lemma 5. *Under the same assumptions of Theorem 2, for all $\delta > 0$, $\exists C_\delta > 0$ such that*

$$\mathbb{E}\left[W_1(Q, Q_n)\right] \leqslant C_\delta n^{-\frac{1}{d+\delta}}. \tag{13}$$

This follows directly from Theorem 1 from [Weed and Bach, 2019]. We attach a full proof in Section B.2 of the appendix. Finally, we prove our statistical estimation result in Theorem 2.

Proof of Theorem 2. Choose $\delta > 0$. Recall from Lemma 4 we have

$$W_1((\hat{g}_n)_\sharp\rho, Q) \leqslant \mathbb{E}\inf_{g_\theta\in\mathcal{G}_{\text{NN}}} W_1((g_\theta)_\sharp\rho, Q) + 2\mathbb{E}W_1(Q_n, Q)$$

The first term is the approximation error which can be controlled within an arbitrarily small accuracy ϵ. Theorem 1 shows the existence of a neural network function g_θ with $O\left(\log\left(\frac{1}{\epsilon}\right)\right)$ layers and $O(D\epsilon^{-d/\alpha}\log(\frac{1}{\epsilon}))$ neurons such that $W_1((g_\theta)_\sharp\rho, Q) \leqslant \epsilon$ for any $\epsilon > 0$. We choose $\epsilon = n^{-\frac{1}{d+\delta}}$ to optimally balance the approximation error and the statistical error. The second term is the statistical error for which we recall from Lemma 5 that $\mathbb{E}\left[W_1(\hat{Q}_n, Q)\right] \leqslant C_\delta n^{-\frac{1}{d+\delta}}$ for some constant C_δ.

Thus we have

$$\mathbb{E}W_1((\hat{g}_n)_\sharp\rho, Q) \leqslant n^{-\frac{1}{d+\delta}} + 2C_\delta n^{-\frac{1}{d+\delta}} = Cn^{-\frac{1}{d+\delta}}$$

by setting $C = 1 + 2C_\delta$. This concludes the proof. □

We remark the above proof proceeds similarly in the noisy case, which is presented in Section B.3 of the appendix.

5 Conclusion

We have established approximation and statistical estimation theories of deep generative models for estimating distributions on a low-dimensional manifold. The statistical convergence rate in this paper depends on the intrinsic dimension of data. In light of the manifold hypothesis, which suggests many natural datasets lie on low dimensional manifolds, our theory rigorously explains why deep generative models defy existing theoretical sample complexity estimates and the curse of dimensionality. In fact, deep generative models are able to learn low-dimensional geometric structures of data, and allow for highly efficient sample complexity independent of the ambient dimension. Meanwhile the size of the required network scales exponentially with the intrinsic dimension.

Our theory imposes very little assumption on the target density Q, requiring only that it admit a density q with respect to the volume measure and that q is upper and lower bounded. In particular we make no smoothness assumptions on q. This is practical, as we do not expect existing natural datasets to exhibit high degrees of smoothness.

In this work, we assume access to computation of the W_1 distance. However during GAN training a discriminator is trained for this purpose. It would be of interest for future work to investigate the low-dimensional role of such discriminator networks which approximate the W_1 distance in practice.

References

Eddie Aamari and Clément Levrard. Nonasymptotic rates for manifold, tangent space and curvature estimation. *The Annals of Statistics*, 2019.

Eddie Aamari, Jisu Kim, Frédéric Chazal, Bertrand Michel, Alessandro Rinaldo, Larry Wasserman, et al. Estimating the reach of a manifold. *Electronic Journal of Statistics*, 13(1):1359–1399, 2019.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (gans). *arXiv preprint arXiv:1703.00573*, 2017.

Yu Bai, Tengyu Ma, and Andrej Risteski. Approximability of discriminators implies diversity in gans. *arXiv preprint arXiv:1806.10586*, 2018.

Adam Block, Zeyu Jia, Yury Polyanskiy, and Alexander Rakhlin. Intrinsic dimension estimation. *arXiv preprint arXiv:2106.04018*, 2021.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

Luis A Caffarelli. The regularity of mappings with a convex potential. *Journal of the American Mathematical Society*, 5(1):99–104, 1992.

Minwoo Chae, Dongha Kim, Yongdai Kim, and Lizhen Lin. A likelihood approach to non-parametric estimation of a singular distribution using deep generative models. *arXiv preprint arXiv:2105.04046*, 2021.

Minshuo Chen, Haoming Jiang, Wenjing Liao, and Tuo Zhao. Nonparametric regression on low-dimensional manifolds using deep relu networks. *arXiv: Learning*, 2019.

Minshuo Chen, Wenjing Liao, Hongyuan Zha, and Tuo Zhao. Statistical guarantees of generative adversarial networks for distribution estimation. *CoRR*, abs/2002.03938, 2020. URL `https://arxiv.org/abs/2002.03938`.

Ronald R Coifman, Stephane Lafon, Ann B Lee, Mauro Maggioni, Boaz Nadler, Frederick Warner, and Steven W Zucker. Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps. *Proceedings of the national academy of sciences*, 102(21):7426–7431, 2005.

L. C. Evans and R. F. Gariepy. *Measure theory and fine properties of functions*. 1992.

Herbert Federer. Curvature measures. *Transactions of the AMS*, pages 418–494, 1959.

F. Flaherty and M.P. do Carmo. *Riemannian Geometry*. Mathematics: Theory & Applications. Birkhäuser Boston, 2013. ISBN 9780817634902. URL `https://books.google.com/books?id=ct91XCWkWEUC`.

Charlie Frogner, Chiyuan Zhang, Hossein Mobahi, Mauricio Araya, and Tomaso A Poggio. Learning with a wasserstein loss. *Advances in neural information processing systems*, 28, 2015.

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

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. 2014.

Jian Huang, Yuling Jiao, Zhen Li, Shiao Liu, Yang Wang, and Yunfei Yang. An error analysis of generative adversarial networks for learning distributions. *Journal of Machine Learning Research*, 23(116):1–43, 2022.

Haoming Jiang, Zhehui Chen, Minshuo Chen, Feng Liu, Dingding Wang, and Tuo Zhao. On computation and generalization of gans with spectrum control. *arXiv preprint arXiv:1812.10912*, 2018.

DP Kingma and M Welling. Auto-encoding variational bayes. iclr 2014 2014. *arXiv preprint arXiv:1312.6114*, 2013.

Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

Tengyuan Liang. How well can generative adversarial networks learn densities: A nonparametric view. *arXiv preprint arXiv:1712.08244*, 2017.

Tengyuan Liang. On how well generative adversarial networks learn densities: Nonparametric and parametric results. *arXiv preprint arXiv:1811.03179*, 2018.

Yulong Lu and Jianfeng Lu. A universal approximation theorem of deep neural networks for expressing probability distributions. *Advances in neural information processing systems*, 33: 3094–3105, 2020.

Giulia Luise, Massimiliano Pontil, and Carlo Ciliberto. Generalization properties of optimal transport gans with latent distribution learning. *arXiv preprint arXiv:2007.14641*, 2020.

Shakir Mohamed and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. 2014.

Alfred Müller. Integral probability metrics and their generating classes of functions. *Advances in Applied Probability*, 29:429–443, 1997. ISSN 00018678. URL `http://www.jstor.org/stable/1428011`.

Ilsang Ohn and Yongdai Kim. Smooth function approximation by deep neural networks with general activation functions. *Entropy*, 21(7):627, 2019.

Gabriel Peyré. Manifold models for signals and images. *Computer vision and image understanding*, 113(2):249–260, 2009.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016.

Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.

Kevin Schawinski, Ce Zhang, Hantian Zhang, Lucas Fowler, and Gokula Krishnan Santhanam. Generative adversarial networks recover features in astrophysical images of galaxies beyond the deconvolution limit. *Monthly Notices of the Royal Astronomical Society: Letters*, 467(1): L110–L114, 2017.

Nicolas Schreuder, Victor-Emmanuel Brunel, and Arnak Dalalyan. Statistical guarantees for generative models without domination. In *Algorithmic Learning Theory*, pages 1051–1071. PMLR, 2021.

Zuowei Shen, Haizhao Yang, and Shijun Zhang. Optimal approximation rate of relu networks in terms of width and depth. *Journal de Mathématiques Pures et Appliquées*, 157:101–135, 2022.

Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

Ananya Uppal, Shashank Singh, and Barnaás Póczos. Nonparametric density estimation & convergence of gans under besov ipm losses. *arXiv preprint arXiv:1902.03511*, 2019.

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

Vanessa Volz, Jacob Schrum, Jialin Liu, Simon M Lucas, Adam Smith, and Sebastian Risi. Evolving mario levels in the latent space of a deep convolutional generative adversarial network. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 221–228, 2018.

Jonathan Weed and Francis R. Bach. Sharp asymptotic and finite-sample rates of convergence of empirical measures in wasserstein distance. *Bernoulli*, 2019.

Yunfei Yang, Zhen Li, and Yang Wang. On the capacity of deep generative networks for approximating distributions. *Neural Networks*, 145:144–154, 2022.

Dmitry Yarotsky. Error bounds for approximations with deep relu networks. *Neural networks : the official journal of the International Neural Network Society*, 94:103–114, 2017.

Pengchuan Zhang, Qiang Liu, Dengyong Zhou, Tao Xu, and Xiaodong He. On the discrimination-generalization tradeoff in gans. *arXiv preprint arXiv:1711.02771*, 2017.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See our conclusion.
   (c) Did you discuss any potential negative societal impacts of your work? [No] We dont foresee any negative societal impact.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] They are attached in the appendix
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [N/A]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Deep ReLU Approximation of Hölder functions

In this section, $\log$ denotes the base 2 logarithm by default. $\times_{i=1}^{d}$ denotes the Cartesian product of d sets. The goal is to determine the approximation rate of deep ReLU networks for Hölder continuous functions. Let $f \in C^{\alpha}([0,1]^d)$ with Hölder norm $\|f\|_{C^\alpha}$ where $\alpha \in (0,1)$. We first approximate f by a piecewise constant function f^n in Section A.1, and then approximate f^n by a deep ReLU Network Φ in Section A.3.

A.1 Piecewise constant approximation

Let $[n]^d = \{(a_1, a_2, \ldots, a_d) : a_i \in \mathbb{N}, 1 \leqslant a_i \leqslant n\}$. Given any $n \in \mathbb{N}$, we cover $[0,1]^d$ by n^d non-overlapping open cubes. For any $(k_1, \ldots, k_d) = \vec{k} \in [n]^d$, we define

$$Q_{\vec{k}} = \bigtimes_{i=1}^{d} \left(\frac{k_i - 1}{n}, \frac{k_i}{n} \right). \tag{14}$$

Lemma 6. *Let $f \in C^{\alpha}([0,1]^d)$ with Hölder norm $\|f\|_{C^\alpha}$. For any $n \in \mathbb{N}$, define*

$$f^n(x) = \sum_{k \in [n]^d} \left(n^d \int_{Q_{\vec{k}}} f(y)\, dy \right) \mathbb{1}_{Q_{\vec{k}}}(x).$$

Then

$$\|f - f^n\|_{L^1} < \|f\|_{C^\alpha} \frac{d^{\alpha/2}}{n^\alpha}.$$

Proof. We estimate

$$\begin{aligned}
\|f - f^n\|_{L^1} &= \int |f(x) - f^n(x)|\, dx \\
&= \int \left| \sum_{k \in [n]^d} \left(n^d \int_{Q_{\vec{k}}} f(x)\, dy \right) \mathbb{1}_{Q_{\vec{k}}}(x) - \sum_{k \in [n]^d} \left(n^d \int_{Q_{\vec{k}}} f(y)\, dy \right) \mathbb{1}_{Q_{\vec{k}}}(x) \right| dx \\
&= n^d \int \left| \sum_{k \in [n]^d} \left(\int_{Q_{\vec{k}}} (f(x) - f(y))\, dy \right) \mathbb{1}_{Q_{\vec{k}}}(x) \right| dx \\
&\leqslant n^d \sum_{k \in [n]^d} \int \mathbb{1}_{Q_{\vec{k}}}(x) \int_{Q_{\vec{k}}} |f(x) - f(y)|\, dy\, dx \\
&= n^d \sum_{k \in [n]^d} \int_{Q_{\vec{k}}} \int_{Q_{\vec{k}}} |f(x) - f(y)|\, dy\, dx \\
&\leqslant n^d \sum_{k \in [n]^d} \int_{Q_{\vec{k}}} \int_{Q_{\vec{k}}} \|f\|_{C^\alpha} \frac{d^{\alpha/2}}{n^\alpha}\, dy\, dx \\
&= \|f\|_{C^\alpha} \frac{d^{\alpha/2}}{n^\alpha} n^d \sum_{k \in [n]^d} \frac{1}{n^{2d}} = \|f\|_{C^\alpha} \frac{d^{\alpha/2}}{n^\alpha}.
\end{aligned}$$

where we use crucially use the fact that $\sup_{x,y \in Q_{\vec{k}}} |f(x) - f(y)| \leqslant \|f\|_{C^\alpha} \sup_{x,y \in Q_{\vec{k}}} |x - y|^\alpha = \|f\|_{C^\alpha} \frac{d^{\alpha/2}}{n^\alpha}$. □

A.2 Neural network approximation

We start with the well-known result originally stated in Yarotsky [2017].

Lemma 7. *Let $A > 0$. For any $\epsilon \in (0, A^2)$, there is a ReLU network which implements a function $\tilde{\times} : \mathbb{R}^2 \to \mathbb{R}$ such that*

$$\sup_{|x| \leqslant A, |y| \leqslant A} \left| \tilde{\times}(x, y) - xy \right| = \epsilon.$$

This network has depth at most $c \log\left(\frac{A^2}{\epsilon}\right)$, width at most 8, and weights bounded by A (where c is an absolute constant).

Proof. The result follows from a careful reading of the proof in Appendix A.2 in Chen et al. [2019]. □

The network given by Lemma 7 approximates the multiplication of two numbers. We seek an approximation of the multiplication of d numbers, and this is achieved by composing $\tilde{\times}$ with itself.

Lemma 8. *Fix $d \in \mathbb{N}$ and let $M > 0$. For any $\epsilon \in (0, M^2)$, there is a ReLU network which implements a function $\tilde{\times}_d : \mathbb{R}^d \to \mathbb{R}$ such that*

$$\sup_{|x_1|, \ldots, |x_d| \leqslant M} \left| \tilde{\times}(x_1, \ldots, x_d) - x_1 \cdots x_d \right| < \epsilon.$$

This network has depth at most $c_1 \log\left(\frac{d^3 M^d}{\epsilon}\right) + c_2$, width at most $8d$, and weights bounded by $2M$ (where c_1 and c_2 are absolute constants).

Proof. Our idea is to realize the multiplication in a binary tree structure, illustrated in Figure 4. We first assume that $2^{k-1} < d \leqslant 2^k$ for some integer k, and let $\delta = \frac{\epsilon}{4^{k-1} M^{2^k - 2}}$. We first handle the case that $d = 2^k$. We will construct a family of k functions $\left\{ \tilde{\times}_{2^i} : \mathbb{R}^{2^i} \to \mathbb{R} \right\}_{i=1}^{k}$ iteratively. We will show that for all $1 \leqslant i \leqslant k$, the function $\tilde{\times}_{2^i}$ implements 2^i-ary multiplication with error at most $4^{i-1} M^{2^i - 2} \delta$ (when $|x_i| < M$), at most $c \log\left(\frac{M^{2^{i+1}-2}}{\delta^i}\right)$ layers, width at most $4 \cdot 2^i$, and weights bounded by M.

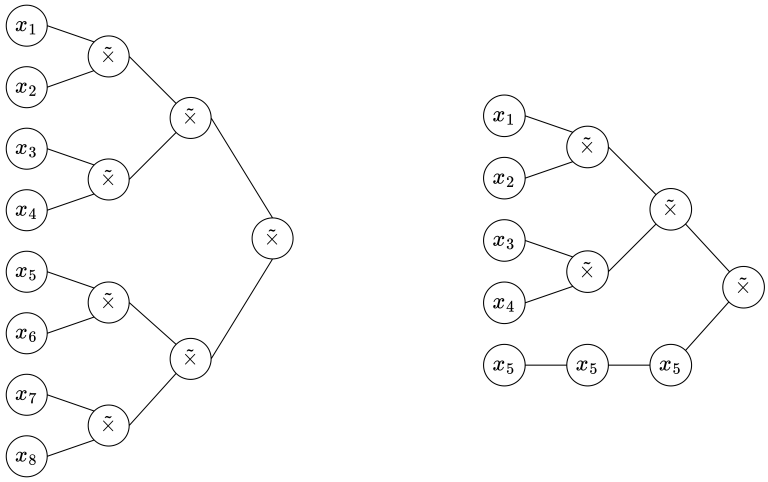
For $i = 1$, we define $\tilde{\times}_2$ to be the function defined in Lemma 7 with the parameters $\epsilon = \delta$ and $A = M$. Then $\tilde{\times}_2$ has maximum error $\delta = 4^{1-1} M^{2^1 - 2} \delta$ and is implementable by a ReLU network with at most $c \log\left(\frac{A^2}{\epsilon}\right) = c \log\left(\frac{M^2}{\delta}\right) = c \log\left(\frac{M^{2^{1+1}-2}}{\delta^1}\right)$ layers, width $8 = 4 \cdot 2^1$, and weights bounded by M, all as desired.

Now suppose the claim has been proven for $\tilde{\times}_{2^i}$. Let $\tilde{\times}$ be the function defined in Lemma 7 with the parameters $\epsilon = \delta$ and $A = M^{2^i}$. Then $\tilde{\times}$ has $c \log\left(\frac{A^2}{\epsilon}\right) = c \log\left(\frac{M^{2^{i+1}}}{\delta}\right)$ layers, width 8, and weights bounded by M. We define

$$\tilde{\times}_{2^{i+1}}(x_1, \ldots, x_{2^{i+1}}) = \tilde{\times}\left(\tilde{\times}_{2^i}(x_1, \ldots, x_{2^i}), \tilde{\times}_{2^i}(x_{2^i+1}, \ldots, x_{2^{i+1}})\right).$$

Then $\tilde{\times}_{2^{i+1}}$ has depth $c \log\left(\frac{M^{2^{i+1}-2}}{\delta^i}\right) + c \log\left(\frac{M^{2^{i+1}}}{\delta}\right) = c \log\left(\frac{M^{2^{i+2}-2}}{\delta^{i+1}}\right)$, width $4 \cdot 2^i + 4 \cdot 2^i = 4 \cdot 2^{i+1}$, and weights bounded by M. It remains to compute the following error bound:

$$\begin{aligned}
&|\tilde{\times}\left(\tilde{\times}_{2^i}(x_1, \ldots, x_{2^i}), \tilde{\times}_{2^i}(x_{2^i+1}, \ldots, x_{2^{i+1}})\right) - x_1 \cdots x_{2^{i+1}}| \\
&\leqslant |\tilde{\times}\left(\tilde{\times}_{2^i}(x_1, \ldots, x_{2^i}), \tilde{\times}_{2^i}(x_{2^i+1}, \ldots, x_{2^{i+1}})\right) - \tilde{\times}_{2^i}(x_1, \ldots, x_{2^i}) \cdot \tilde{\times}_{2^i}(x_{2^i+1}, \ldots, x_{2^{i+1}})| \\
&\quad + |\tilde{\times}_{2^i}(x_1, \ldots, x_{2^i}) \cdot \tilde{\times}_{2^i}(x_{2^i+1}, \ldots, x_{2^{i+1}}) - \tilde{\times}_{2^i}(x_1, \ldots, x_{2^i}) \cdot x_{2^i+1} \cdots x_{2^{i+1}}| \\
&\quad + |\tilde{\times}_{2^i}(x_1, \ldots, x_{2^i}) \cdot x_{2^i+1} \cdots x_{2^{i+1}} - x_1 \cdots x_{2^{i+1}}| \\
&\leqslant \delta + |\tilde{\times}_{2^i}(x_1, \ldots, x_{2^i})| \cdot |\tilde{\times}_{2^i}(x_{2^i+1}, \ldots, x_{2^{i+1}}) - x_{2^i+1} \cdots x_{2^{i+1}}| \\
&\quad + |x_{2^i+1} \ldots x_{2^{i+1}}| \cdot |\tilde{\times}_{2^i}(x_1, \ldots, x_{2^i}) - x_1 \cdots x_{2^i}| \\
&\leqslant \delta + M^{2^i}(4^{i-1} M^{2^i - 2} \delta) + M^{2^i}(4^{i-1} M^{2^i - 2} \delta)\delta \\
&= (1 + 24^{i-1} M^{2^{i+1}-2})\delta \\
&< 4^i M^{2^{i+1}-2} \delta.
\end{aligned}$$



(a) $\tilde{\times}_8$ (b) $\tilde{\times}_5$

Figure 4: Network diagrams for ReLU networks approximating multiplication in Lemma 8.

From this, we have constructed a function $\tilde{\times}_{2^k}$ that approximates multiplication (of values $< M$) with error at most $4^{k-1}M^{2^k-2}\delta = \epsilon$ that has depth

$$\begin{aligned}
c\log\left(\frac{M^{2^{k+1}-2}}{\delta^k}\right) &= c\log\left(\frac{M^{2^k-2}M^{2^k}(4^{k-1})^k(M^{2^k-2})^k}{\epsilon^k}\right) \\
&= c\log\left(M^{2^k-2}\right) + c\log\left(\frac{(4^{k-1})^k(M^{2^k})^k}{\epsilon^k}\right) \\
&= c\log\left(M^{2^k-2}\right) + ck\log\left(\frac{4^{k-1}M^{2^k}}{\epsilon}\right) \\
&< c(1+k)\log\left(\frac{4^{k-1}M^{2^k}}{\epsilon}\right) \\
&< c_1 k\log\left(\frac{4^{k-1}M^{2^k}}{\epsilon}\right),
\end{aligned}$$

For some absolute constant c_1. Now since $k = \log(d)$, we have that $M^{2^k} = M^d$ and $4^{k-1} < 4^k = d^2$, so the ReLU network has depth at most $c\log(d)\log\left(\frac{d^2M^d}{\epsilon}\right)$ where c is an absolute constant (the same constant as in Lemma 7). The width of $\tilde{\times}_{2^k}$ is $4 \cdot 2^k = 4d$, and the weights are bounded by M.

Figure 4(a) shows a neural network diagram for the ReLU network implementing $\tilde{\times}_8$, which has the structure of a full binary tree. In order to handle numbers that are not powers of two, we use an architecture similar to the diagram in Figure 4 (b) which depicts the ReLU network implementing $\tilde{\times}_5$.

Formally, suppose we have $2^{i-1} < d \leqslant 2^i$ for some $i \in \mathbb{N}$. Then consider the network $\tilde{\times}_{2^i}$ defined as before, but we remove the last $2^i - d$ input neurons, and replace them with 1 everywhere they appear. Note that this can be achieved by adjusting the bias of each neuron appropriately. For example, any neuron can be turned into a constant 1 by making the weight vector 0 and making the bias equal to 1. This procedure will not affect the number of layers, it will not increase the width, and the parameters are bounded by $M + 1 < 2M$. Noting that $2^i < 2d$, we see that the ReLU network has width at most $4 \cdot 2^i < 4 \cdot 2d = 8d$. Finally, the depth is at most (for c_1 and c_2 absolute constants)

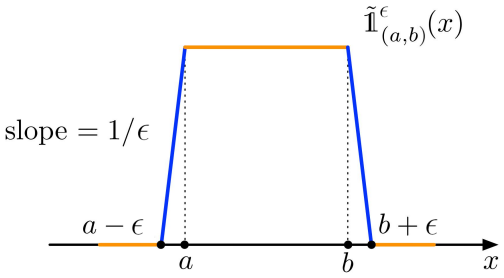


Figure 5: Plot of $\tilde{\mathbb{1}}^{\epsilon}_{(a,b)}$.

$$
\begin{aligned}
c\log(2^i)\log\left(\frac{(2^i)^2M^d}{\epsilon}\right) &< c\log(2d)\log\left(\frac{(2d)^2M^d}{\epsilon}\right)\\
&= c(\log(2)+\log(d))\left(\log(4)+\log\left(\frac{d^2M^d}{\epsilon}\right)\right)\\
&< c\left(\log(8)+3\log(d)\log\left(\frac{d^2M^d}{\epsilon}\right)\right)\\
&= c_1\log\left(\frac{d^3M^d}{\epsilon}\right)+c_2.
\end{aligned}
$$

□

Next we approximate the indicator functions of intervals (which we denote by $\mathbb{1}_{(a,b)}$).

Lemma 9. *Fix $M > 1$. Let $[a,b] \subseteq [-M,M]$ and $\epsilon < \frac{1}{2}(b-a)$. Then there is a ReLU network which implements a function $\tilde{\mathbb{1}}^{\epsilon}_{(a,b)}$ such that*

$$\left\|\tilde{\mathbb{1}}^{\epsilon}_{(a,b)} - \mathbb{1}_{(a,b)}\right\|_{L^1} = \epsilon.$$

This network has depth at most $c\log\left(\frac{1}{\epsilon}\right)$, width equal to 4, and weights bounded by M (where c is a constant depending only on M).

Proof. We define the ReLU network function

$$\tilde{\mathbb{1}}^{\epsilon}_{(a,b)}(x) = \frac{1}{\epsilon}(\sigma(x-(a-\epsilon)) - \sigma(x-a) - \sigma(x-b) + \sigma(x-(b+\epsilon))),$$

where σ is the ReLU activation function. Figure 5 is a plot of $\tilde{\mathbb{1}}^{\epsilon}_{(a,b)}$. Then it is clear that

$$\left\|\tilde{\mathbb{1}}^{\epsilon}_{(a,b)} - \mathbb{1}_{(a,b)}\right\|_{L^1} = \epsilon.$$

Note that we can express $\frac{1}{\epsilon}$ as

$$\frac{1}{\epsilon} = M^{\lceil\log_M(\frac{1}{\epsilon})\rceil-1}\cdot\frac{\frac{1}{\epsilon}}{M^{\lceil\log_M(\frac{1}{\epsilon})\rceil-1}} = \underbrace{M\times M\times\cdots\times M}_{\lceil\log_M(\frac{1}{\epsilon})\rceil-1 \text{ times}}\times R,$$

where $0 < R \leqslant M$. This is a product of $\left\lceil\log_M\left(\frac{1}{\epsilon}\right)\right\rceil$ numbers that are all bounded by M. Then note that

$$\tilde{\mathbb{1}}^{\epsilon}_{(a,b)} = M\times\cdots\times M\times R(\sigma(x-(a-\epsilon)) - \sigma(x-a) - \sigma(x-b) + \sigma(x-(b+\epsilon)))$$

can be implemented by ReLU network with $1+\left\lceil\log_M\left(\frac{1}{\epsilon}\right)\right\rceil$ layers. The first layer has 4 neurons, and the second layer has one neuron, and they together compute $R(\sigma(x-(a-\epsilon)) - \sigma(x-a) - \sigma(x-$

$b) + \sigma(x - (b + \epsilon)))$. Then the next $\left\lceil \log_M \left(\frac{1}{\epsilon}\right)\right\rceil - 1$ each multiply this value by M (since all values at this point are positive, the ReLU activation does nothing at each layer). Thus the ReLU network has width 4 (though only the first layer has more than one neuron) and weights bounded by M. $\square$

We combine Lemma 8 and Lemma 9 to obtain an approximation to the indicator function of d-dimensional cube.

Lemma 10. *Fix $M > 1$. Let $Q = \times_{k=1}^{d}(a_k, b_k) \subseteq [-M, M]^d$ be a bounded d-dimensional cube (i.e. $b_1 - a_1 = b_k - a_k$ for all $k \in [d]$), and suppose $\epsilon < \min\left(\frac{b_1 - a_1}{2}, 1\right)$. Then there exists a function $\phi : [-M, M]^d \to \mathbb{R}$ implementable by a ReLU network such that*

$$\int_{[-M,M]^d} |\mathbb{1}_Q(x) - \phi(x)|\, dx < \epsilon.$$

The network has depth at most $c_1 \log\left(\frac{d^2 4^d}{\epsilon}\right) + c_2$, width at most $4d$, and weights bounded by $\max\{M, 2\}$ (where c_1 and c_2 are constants only depending on M).

Proof. Denote by $\tilde{\mathbb{1}}^{\delta}_{(a_i,b_i)}$ the approximation to $\mathbb{1}_{(a_i,b_i)}$ obtained from Lemma 9 with $\delta = \frac{\epsilon}{2}$. Let $\eta = \frac{\epsilon}{2^{d+1}M^d}$, and denote by $\tilde{\times}_d$ the approximation of the multiplication of d factors obtained from Lemma 8 with parameters $M = 1$ and error η (which is denoted as ϵ in the lemma statement). Then we define ϕ by

$$\phi(x_1, \ldots, x_d) = \tilde{\times}_d \left(\tilde{\mathbb{1}}^{\delta}_{(a_1,b_1)}(x_1), \ldots, \tilde{\mathbb{1}}^{\delta}_{(a_d,b_d)}(x_d)\right).$$

We compute

$$\begin{aligned}
&\int_{[-M,M]^d} \left|\mathbb{1}_{\times_{i=1}^{d}[a_i,b_i)}(x) - \phi(x)\right| dx \\
&= \int_{[-M,M]^d} \left|\prod_{i=1}^{d} \mathbb{1}_{(a_i,b_i)}(x_i) - \tilde{\times}_d \left(\tilde{\mathbb{1}}^{\delta}_{(a_1,b_1)}(x_1), \ldots, \tilde{\mathbb{1}}^{\delta}_{(a_d,b_d)}(x_d)\right)\right| dx \\
&\leqslant \int_{[-M,M]^d} \left|\prod_{i=1}^{d} \mathbb{1}_{(a_i,b_i)}(x_i) - \prod_{i=1}^{d} \tilde{\mathbb{1}}^{\delta}_{(a_i,b_i)}(x_i)\right| dx \\
&\qquad + \int_{[-M,M]^d} \left|\prod_{i=1}^{d} \tilde{\mathbb{1}}^{\delta}_{(a_i,b_i)}(x_i) - \tilde{\times}_d \left(\tilde{\mathbb{1}}^{\delta}_{(a_1,b_1)}(x_1), \ldots, \tilde{\mathbb{1}}^{\delta}_{(a_d,b_d)}(x_d)\right)\right| dx \\
&< \int_{\times_{i=1}^{d}[a_i-\delta,b_i+\delta] \setminus \times_{i=1}^{d}[a_i,b_i)} \left|\prod_{i=1}^{d} \tilde{\mathbb{1}}^{\delta}_{(a_i,b_i)}(x_i)\right| + \int_{[-M,M]^d} \frac{\epsilon}{2^{d+1}M^d}\, dx \\
&< \mathrm{Vol}\left(\bigtimes_{i=1}^{d}(a_i - \delta, b_i + \delta) \setminus \bigtimes_{i=1}^{d}(a_i, b_i)\right) + \frac{\epsilon}{2}
\end{aligned}$$

where last inequality follows since $\left|\prod_{i=1}^{d} \tilde{\mathbb{1}}^{\delta}_{(a_i,b_i)}(x_i)\right| < 1$. We now focus on bounding the measure of $R = \left(\times_{i=1}^{d}(a_i - \delta, b_i + \delta) \setminus \times_{i=1}^{d}(a_i, b_i)\right)$. First we express $R = \bigcup_{k=1}^{d} S_k^+ \cup \bigcup_{k=1}^{d} S_k^-$ where

$$S_k^- = \left(\bigtimes_{i=1}^{k-1}(a_i - \delta, b_i + \delta)\right) \times (a_k - \delta, a_k) \times \left(\bigtimes_{i=k+1}^{d}(a_i - \delta, b_i + \delta)\right),$$

$$S_k^+ = \left(\bigtimes_{i=1}^{k-1}(a_i - \delta, b_i + \delta)\right) \times (b_k, b_k + \delta) \times \left(\bigtimes_{i=k+1}^{d}(a_i - \delta, b_i + \delta)\right).$$

Thus we have that $\text{Vol}(R) \leqslant \sum_{k=1}^{d} \text{Vol}(S_k^+) + \text{Vol}(S_k^-) = 2(2\delta + b_i - a_i)^{d-1}\delta < 2(2\delta + 2M)^{d-1}\delta$. If we pick $\delta = \frac{\epsilon}{4(3M)^{d-1}}$, then we have

$$\begin{aligned}
\|\mathbb{1}_Q - \phi\|_{L^1} &< \text{Vol}\left(\bigtimes_{i=1}^{d}(a_i - \delta, b_i + \delta) \setminus \bigtimes_{i=1}^{d}(a_i, b_i)\right) + \frac{\epsilon}{2} \\
&\leqslant 2(2\delta + 2M)^{d-1}\delta + \frac{\epsilon}{2} \\
&= 2(3M)^{d-1}\frac{\epsilon}{4(3M)^{d-1}} + \frac{\epsilon}{2} \\
&= \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon.
\end{aligned}$$

Finally, we determine the size of ϕ. Each $\tilde{\mathbb{1}}^{\delta}_{(a_i,b_i)}$ can be implemented by a ReLU network with depth $c_1 \log\left(\frac{1}{\delta}\right) = c_1 \log\left(\frac{4 \cdot 3^{d-1} M^{d-1}}{\epsilon}\right) \leqslant c_2 \log\left(\frac{3^{d-1}M^{d-1}}{\epsilon}\right)$, width 4, and weights bounded by M. $\tilde{\times}_d$ can be implemented by a ReLU network with depth at most

$$c_3 \log\left(\frac{d^3 2^d}{\eta}\right) + c_4 = c_3 \log\left(\frac{d^3 2^d 2^{d+1} M^d}{\epsilon}\right) + c_4 = c_3 \log\left(\frac{d^3 2^{2d+1} M^d}{\epsilon}\right) + c_4,$$

width at most 4d, and weights bounded by 2. Thus ϕ has width at most $4d$, weights bounded by $\max\{M, 2\}$, and depth at most

$$\begin{aligned}
&c_2 \log\left(\frac{3^{d-1}M^{d-1}}{\epsilon}\right) + c_3 \log\left(\frac{d^3 2^{2d+1} M^d}{\epsilon}\right) + c_4 \\
&< \max(c_2, c_3) \log\left(\frac{d^3 3^{d-1} 2^{2d+1} M^{2d-1}}{\epsilon^2}\right) + c_4 \\
&< \max(c_2, c_3) \log\left(\frac{d^3 4^{d-1} 4^{d+1} M^{2d}}{\epsilon^2}\right) + c_4 \\
&< \max(c_2, c_3) \log\left(\frac{(d^2 4^d M^d)^2}{\epsilon^2}\right) + c_4 \\
&= c_5 \log\left(\frac{d^2 4^d M^d}{\epsilon}\right) + c_4 \\
&= c_5 \log\left(\frac{d^2 4^d}{\epsilon}\right) + c_5 d \log(M) + c_4. \\
&< c_6 \log\left(\frac{d^2 4^d}{\epsilon}\right) + c_4.
\end{aligned}$$

□

Now we use the construction of Lemma 10 to approximate the function from Lemma 6 as follows. Let $\{Q_{\vec{k}}\}_{\vec{k} \in [n]^d}$ be a decomposition of $[0,1]^d$ into almost non-overlapping cubes as in Lemma 6. The only overlap of these cubes is a set of measure 0.

Lemma 11. *Let $0 < \epsilon < \frac{1}{2n}$. Let $\{\beta_{\vec{k}}\}_{\vec{k} \in [n]^d}$ be constants within $[-M, M]$. Then the function g defined by*

$$g(x) = \sum_{k \in [n]^d} \beta_{\vec{k}} \mathbb{1}_{Q_{\vec{k}}}(x)$$

can be approximated by a neural network $\tilde{g}$ with depth $c_1 \log\left(\frac{d^2 4^d n^d}{\epsilon}\right) + c_2$, width $4dn^d$, and weights bounded by $\max\{M, 2\}$ (where c_1 and c_2 are constants only depending on M), such that

$$\int_{[-M,M]^d} |g(x) - \tilde{g}(x)|\, dx < \epsilon.$$

Proof. For every $\vec{k} \in [n]^d$, let $\phi_{\vec{k}}$ be the function from Lemma 10 that approximates $\mathbb{1}_{Q_{\vec{k}}}$ with error $\frac{\epsilon}{Mn^d}$. Then each $\phi_{\vec{k}}$ can be implemented by a ReLU network with

$$c_1 \log\left(\frac{d^2 4^d n^d M}{\epsilon}\right) + c_2 = c_1 \log\left(\frac{d^2 4^d n^d}{\epsilon}\right) + c_3$$

layers, width at most $4d$, and weights bounded by 2. This means we can implement the function $\tilde{g}(x) = \sum_{k\in[n]^d} \beta_{\vec{k}} \phi_{\vec{k}}$ by a ReLU network with one more layer, width at most $4dn^d$, and weights bounded by $\max\{M, 2\}$. We compute

$$\begin{aligned}
\int_{[-M,M]^d} |g(x) - \tilde{g}(x)|\, dx &= \int_{[-M,M]^d} \left| \sum_{k\in[n]^d} \beta_{\vec{k}} \mathbb{1}_{Q_{\vec{k}}}(x) - \sum_{k\in[n]^d} \beta_{\vec{k}} \phi_{\vec{k}}(x) \right| dx \\
&\leqslant \sum_{k\in[n]^d} |\beta_{\vec{k}}| \int_{[-M,M]^d} \left| \mathbb{1}_{Q_{\vec{k}}}(x) - \phi_{\vec{k}}(x) \right| dx \\
&\leqslant \sum_{k\in[n]^d} |\beta_{\vec{k}}| \left(\frac{\epsilon}{Mn^d}\right) \\
&\leqslant \sum_{k\in[n]^d} M \left(\frac{\epsilon}{Mn^d}\right) = \epsilon.
\end{aligned}$$

□

A.3 Putting approximations together

We combine Lemma 6 with Lemma 11 to obtain our approximation result. The requirement of $d \geqslant 4$ in the following lemma is for technical reasons, and is not a requirement for Lemma 3 which is used in the main paper.

Lemma 12. *Suppose $f \in C^\alpha([0,1]^d)$, $\alpha \in (0,1)$, with $\|f\|_{L^\infty} < M$ and $M \geqslant 2$. Assume further that $d \geqslant 4$. Let $\epsilon > 0$. Then there exists a function Φ implementable by a ReLU network such that*

$$\|f - \Phi\|_{L^1} < \epsilon.$$

The ReLU network has depth at most $c_1 d \log\left(\frac{8d\|f\|_{C^\alpha}^{1/\alpha}}{\epsilon^{2/\alpha}}\right) + c_2$, width at most $\frac{(4d)^{d/2}\|f\|_{C^\alpha}^{d/\alpha}}{\epsilon^{d/\alpha}}$, and weights bounded by M (where c_1 and c_2 are constants only depending on M).

Proof. Let $n = \left\lceil \left(\frac{\|f\|_{C^\alpha}}{\epsilon}\right)^{1/\alpha} \sqrt{d} \right\rceil$. Let f^n be the piecewise constant function from Lemma 6. Notice that f^n follows the same form as the function g in Lemma 11. By Lemma 11, there is a ReLU network function Φ that approximates f^n such that

$$\int_{[-M,M]^d} |f^n(x) - \Phi(x)|\, dx < \frac{\epsilon}{2}.$$

Then we compute

$$\|f - \Phi\|_{L^1} \leqslant \|f - f^n\|_{L^1} + \|f^n - \Phi\|_{L^1} < \frac{\|f\|_{C^\alpha} d^{\alpha/2}}{n^\alpha} + \frac{\epsilon}{2} < \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon.$$

Note that Φ has width $4dn^d < 4d\left(2\left(\frac{\|f\|_{C^\alpha}}{\epsilon}\right)^{1/\alpha}\sqrt{d}\right)^d = c_0\frac{(4d)^{d/2}\|f\|_{C^\alpha}^{d/\alpha}}{\epsilon^{d/\alpha}}$, and the weights of Φ are bounded by M. The depth of Φ is bounded by

$$\begin{aligned}
c_1\log\left(\frac{d^2 4^d n^d}{\epsilon}\right)+c_2 &< c_1\log\left(\frac{d^2 4^d \|f\|_{C^\alpha}^{\frac{d}{\alpha}} 2^d d^{\frac{d}{2}}}{\epsilon\epsilon^{\frac{d}{\alpha}}}\right)+c_2\\
&< c_1\log\left(\frac{d^d 8^d \|f\|_{C^\alpha}^{\frac{d}{\alpha}}}{\epsilon^{\frac{2d}{\alpha}}}\right)+c_2\\
&= c_1 d\log\left(\frac{8d\|f\|_{C^\alpha}^{1/\alpha}}{\epsilon^{2/\alpha}}\right)+c_2
\end{aligned}$$

where we use the fact that $d \geqslant 4$ to bound $2+\frac{d}{2} \leqslant d$.

□

In the main paper, we use Lemma 3, which is proved below.

Proof of Lemma 3. By allowing the constants to depend on all values except ϵ, the depth of the ReLU network Φ from Lemma 12 can be expressed as

$$c_1 d\log\left(\frac{8d\|f\|_{C_\alpha}^{1/\alpha}}{\epsilon^{2/\alpha}}\right)+c_2 = c_3\left(\log\left(\frac{1}{\epsilon^{2/\alpha}}\right)+\log\left(8d\|f\|_{C_\alpha}^{1/\alpha}\right)\right)+c_2 = \frac{2}{\alpha}c_3\log\left(\frac{1}{\epsilon}\right)+c_4 < c_5\log\left(\frac{1}{\epsilon}\right),$$

and the width can be expressed as

$$\frac{(4d)^{d/2}\|f\|_{C^\alpha}^{d/\alpha}}{\epsilon^{d/\alpha}} = \frac{c_6}{\epsilon^{d/\alpha}}.$$

□

A.4 Approximating g^* in (10) with a neural network

The goal is to show that the L^1 distance between g_θ (as defined in 11) and g^* is small. We have

$$\begin{aligned}
&\|g^*-g_\theta\|_{L^1}\\
&=\sum_{i=1}^{D}||(g^*)^{(i)}-(g_\theta)^{(i)}||_{L^1}\\
&=\sum_{i=1}^{D}\int_{(0,1)^{d+1}}\left|(g^*)^{(i)}(x)-(g_\theta)^{(i)}(x)\right|dx\\
&\leqslant\sum_{i=1}^{D}\sum_{j=1}^{J}\int_{(0,1)^{d+1}}\left|\tilde{\times}^{\delta_2}\left(\tilde{\mathbb{1}}^{\delta_1}_{(\pi_{j-1},\pi_j)}(x_1),(g^{\delta_3}_{j,\theta})^{(i)}(x_{2:d+1})\right)-\mathbb{1}_{(\pi_{j-1},\pi_j)}(x_1)(g_j^*)^{(i)}(x_{2:d+1})\right|dx\\
&\leqslant\sum_{i=1}^{D}\sum_{j=1}^{J}\int_{(0,1)^{d+1}}\left|\tilde{\times}^{\delta_2}\left(\tilde{\mathbb{1}}^{\delta_1}_{(\pi_{j-1},\pi_j)}(x_1),(g^{\delta_3}_{j,\theta})^{(i)}(x_{2:d+1})\right)-\tilde{\mathbb{1}}^{\delta_1}_{(\pi_{j-1},\pi_j)}(x_1)(g^{\delta_3}_{j,\theta})^{(i)}(x_{2:d+1})\right|dx\\
&\quad+\sum_{i=1}^{D}\sum_{j=1}^{J}\int_{(0,1)^{d+1}}\left|\tilde{\mathbb{1}}^{\delta_1}_{(\pi_{j-1},\pi_j)}(x_1)(g^{\delta_3}_{j,\theta})^{(i)}(x_{2:d+1})-\mathbb{1}_{(\pi_{j-1},\pi_j)}(x_1)(g^{\delta_3}_{j,\theta})^{(i)}(x_{2:d+1})\right|dx\\
&\quad+\sum_{i=1}^{D}\sum_{j=1}^{J}\int_{(0,1)^{d+1}}\left|\mathbb{1}_{(\pi_{j-1},\pi_j)}(x_1)(g^{\delta_3}_{j,\theta})^{(i)}(x_{2:d+1})-\mathbb{1}_{(\pi_{j-1},\pi_j)}(x_1)(g_j^*)^{(i)}(x_{2:d+1})\right|dx\\
&=\sum_{i=1}^{D}\sum_{j=1}^{J}(\text{(I)}+\text{(II)}+\text{(III)})
\end{aligned}$$

Each of the three terms are easily handled as follows.

(I) By construction of $\tilde{\times}^{\delta_2}$ in Lemma 7, we have that

$$\begin{aligned}\text{(I)} &= \int_{(0,1)^{d+1}} \left| \tilde{\times}^{\delta_2} \left(\tilde{\mathbb{1}}^{\delta_1}_{(\pi_{j-1},\pi_j)}(x_1), (g^{\delta_3}_{j,\theta})^{(i)}(x_{2:d+1}) \right) - \tilde{\mathbb{1}}^{\delta_1}_{(\pi_{j-1},\pi_j)}(x_1)(g^{\delta_3}_{j,\theta})^{(i)}(x_{2:d+1}) \right| dx \\ &\leqslant \int_{(0,1)^{d+1}} \delta_2 \, dx = \delta_2.\end{aligned}$$

(II) By construction of $\tilde{\mathbb{1}}^{\delta_1}_{(\pi_{j-1},\pi_j)}$ in Lemma 9, we have that

$$\begin{aligned}\text{(II)} &= \int_{(0,1)^{d+1}} \left| \tilde{\mathbb{1}}^{\delta_1}_{(\pi_{j-1},\pi_j)}(x_1)(g^{\delta_3}_{j,\theta})^{(i)}(x_{2:d+1}) - \mathbb{1}_{(\pi_{j-1},\pi_j)}(x_1)(g^{\delta_3}_{j,\theta})^{(i)}(x_{2:d+1}) \right| dx \\ &\leqslant \left\| (g^{\delta_3}_{j,\theta})^{(i)} \right\|_\infty \int_0^1 \left| \tilde{\mathbb{1}}^{\delta_1}_{(\pi_{j-1},\pi_j)}(x_1) - \mathbb{1}_{(\pi_{j-1},\pi_j)}(x_1) \right| dx \\ &\leqslant M \left\| \tilde{\mathbb{1}}^{\delta_1}_{(a,b)} - \mathbb{1}_{(a,b)} \right\|_{L^1} \\ &= M\delta_1.\end{aligned}$$

(III) By construction of $(g^{\delta_3}_{j,\theta})^{(i)}$ from Lemma 3, we have that

$$\begin{aligned}\text{(III)} &= \int_{(0,1)^{d+1}} \left| \mathbb{1}_{(\pi_{j-1},\pi_j)}(x_1)(g^{\delta_3}_{j,\theta})^{(i)}(x_{2:d+1}) - \mathbb{1}_{(\pi_{j-1},\pi_j)}(x_1)(g^*_j)^{(i)}(x_{2:d+1}) \right| dx \\ &= ||\mathbb{1}_{(\pi_{j-1},\pi_j)}||_\infty \int_{(0,1)^d} \left| (g^{\delta_3}_{j,\theta})^{(i)}(x) - (g^*_j)^{(i)}(x) \right| dx \\ &= \left\| (g^{\delta_3}_{j,\theta})^{(i)} - (g^*_j)^{(i)} \right\|_{L^1} \\ &\leqslant \delta_3.\end{aligned}$$

As a result, we have that

$$\|g^* - g_\theta\|_{L^1} \leqslant \sum_{i=1}^{D} \sum_{j=1}^{J} \text{(I)} + \text{(II)} + \text{(III)} \leqslant \sum_{i=1}^{D} \sum_{j=1}^{J} \delta_2 + M\delta_1 + \delta_3 = DJ(M\delta_1 + \delta_2 + \delta_3).$$

By selecting $\delta_1 < \frac{\epsilon}{3DJM}$, $\delta_2 < \frac{\epsilon}{3DJ}$, and $\delta_3 < \frac{\epsilon}{3DJ}$, we obtain that $||g^* - g_\theta||_1 < \epsilon$.

To complete the proof, we note that g_θ can be exactly represented by a neural network in $\mathcal{G}_{\text{NN}}(L, p, \kappa)$ with parameters

$$L = O\left(\log\left(\frac{1}{\epsilon}\right)\right), \quad p = O\left(D\epsilon^{-\frac{d}{\alpha}}\right), \quad \kappa = M.$$

B Statistical Lemmas

Here we present proofs of lemmas used in our statistical theory in Section 4.2.

B.1 Distribution approximation in W_1 via function approximation in L^1

Proof of Lemma 2. The vector-valued functions f and g output m-dimensional vectors. Note that $||f - g||_{L^1(\mu)} = \sum_{i=1}^{m} \|f_i - g_i\|$ where f_i and g_i denote the ith component function of f and g,

respectively. Then we can compute

$$\begin{aligned}
W_1(f_\sharp\mu, g_\sharp\mu) &= \sup_{\phi\in \mathrm{Lip}_1(\mathbb{R}^m)} \left| \int \phi(y)\, d(f_\sharp\mu) - \int \phi(y)\, d(g_\sharp\mu) \right| \\
&= \sup_{\phi\in \mathrm{Lip}_1(\mathbb{R}^m)} \left| \int \phi(f(x)) - \phi(g(x))\, d\mu \right| \\
&\leqslant \sup_{\phi\in \mathrm{Lip}_1(\mathbb{R}^m)} \int |\phi(f(x)) - \phi(g(x))|\ d\mu \\
&\leqslant \int_Z \|f(x) - g(x)\|_2\ d\mu \\
&\leqslant \int_Z C\|f(x) - g(x)\|_1\, d\mu \\
&= C\|f - g\|_{L^1(\mu)},
\end{aligned}$$

since ϕ is Lipschitz with constant 1 and all norms are equivalent in finite dimensions. In particular, $C = 1$ here.

□

B.2 Convergence of empirical measure

Proof of Lemma 5. Let $\delta > 0$. Consider the manifold $\mathcal{M}$ with the geodesic distance as a metric space. When [Weed and Bach, 2019, Theorem 1] is applied to $\mathcal{M}$ with the geodesic distance, we have that

$$\mathbb{E}\left[W_1^{\mathcal{M}}(Q, Q_n)\right] \leqslant C_\delta n^{-\frac{1}{d+\delta}}$$

for some constant C_δ independent of n. Here, $W_1^{\mathcal{M}}$ is the 1-Wasserstein distance on $\mathcal{M}$ with the geodesic distance. It suffices to show that

$$W_1^{\mathbb{R}^D}(Q, Q_n) = W_1(Q, Q_n) \leqslant W_1^{\mathcal{M}}(Q, Q_n).$$

Let $\mathrm{Lip}_1(\mathbb{R}^D)$ and $\mathrm{Lip}_1(\mathcal{M})$ denote the set of 1-Lipschitz functions defined on $\mathcal{M}$ with respect to the Euclidean distance on $\mathbb{R}^D$ and geodesic distance on $\mathcal{M}$ respectively. But note that $\mathrm{Lip}_1(\mathbb{R}^D) \subseteq \mathrm{Lip}_1(\mathcal{M})$ because for any $f \in \mathrm{Lip}_1(\mathbb{R}^D)$ we have

$$\frac{|f(x) - f(y)|}{\|x - y\|_{\mathcal{M}}} \leqslant \frac{|f(x) - f(y)|}{\|x - y\|_{\mathbb{R}^D}} \leqslant 1$$

as $\|x - y\|_{\mathbb{R}^D} \leqslant \|x - y\|_{\mathcal{M}}$ under an isometric embedding and hence $f \in \mathrm{Lip}_1(\mathcal{M})$. Thus

$$\mathbb{E}\left[W_1(Q, Q_n)\right] \leqslant \mathbb{E}\left[W_1^{\mathcal{M}}(Q, Q_n)\right] \leqslant C_\delta n^{-\frac{1}{d+\delta}}.$$

□

B.3 Controlling the noisy samples

In the noisy setting, we are given n noisy i.i.d. samples $\hat{X}_1, ..., \hat{X}_n$ of the form $\hat{X}_i = X_i + \xi_i$, for $X_i \sim Q$ and ξ_i distributed according to some noise distribution. The optimization in (4) is performed with the noisy empirical distribution $\hat{Q}_n = \frac{1}{n}\sum_{i=1}^n \delta_{\hat{X}_i}$.

Lemma 13. *Under the same assumptions of Theorem 2 and in the noisy setting, we have*

$$\mathbb{E}W_1((\hat{g}_n)_\sharp\rho, Q) \leqslant \inf_{g_\theta\in\mathcal{G}_{\mathrm{NN}}} W_1((g_\theta)_\sharp\rho, Q) + 2\mathbb{E}W_1(Q_n, Q) + 2\mathbb{E}W_1(\hat{Q}_n, Q_n) \tag{15}$$

where $\hat{Q}_n$ is the noisy empirical distribution and Q_n is the clean empirical distribution.

Proof. We compute recalling the definition of $\hat{g}_n$ as the empirical risk minimizer.

$$\begin{aligned}
\mathbb{E}W_1((\hat{g}_n)_\sharp\rho, Q) &\leqslant \mathbb{E}W_1((\hat{g}_n)_\sharp\rho, \hat{Q}_n) + \mathbb{E}W_1(\hat{Q}_n, Q) \\
&\leqslant \mathbb{E}\inf_{g_\theta\in\mathcal{G}_{\mathrm{NN}}} W_1((g_\theta)_\sharp\rho, \hat{Q}_n) + \mathbb{E}W_1(Q_n, Q) + \mathbb{E}W_1(\hat{Q}_n, Q_n) \\
&\leqslant \mathbb{E}\inf_{g_\theta\in\mathcal{G}_{\mathrm{NN}}} W_1((g_\theta)_\sharp\rho, Q) + 2\mathbb{E}W_1(Q_n, Q) + 2\mathbb{E}W_1(\hat{Q}_n, Q)
\end{aligned}$$

where where we recall $W_1((\hat{g}_n)_\sharp \rho, \hat{Q}_n) = \inf_{g_\theta \in \mathcal{G}_{\mathrm{NN}}} W_1((g_\theta)_\sharp \rho, \hat{Q}_n)$ from (4).

□

Lemma 14. *Write $W_1(\hat{Q}_n, Q_n) = W_1^{\mathbb{R}^D}(\hat{Q}_n, Q_n)$. In the noisy setting, we express $\hat{X}_i = X_i + \xi_i$ where X_i is drawn from Q and then noised with ξ_i drawn from some noise distribution. Then*

$$\mathbb{E}[W_1(Q_n, \hat{Q}_n)] \leqslant \sqrt{V_\xi}$$

where $V_\xi = \mathbb{E}\|\xi\|_2^2$ which is the variance of the noise.

Proof. Let $\hat{X}_i, X_i$ be samples defining $\hat{Q}_n, Q_n$ respectively. We have $\hat{X}_i = X_i + \xi_i$ where ξ is the noise term. Compute

$$\begin{aligned}
\mathbb{E}W_1(\hat{Q}_n, Q_n) &= \mathbb{E} \sup_{f \in \mathrm{Lip}_1(\mathbb{R}^D)} \hat{Q}_n(f) - Q_n(f) = \mathbb{E} \sup_{f \in \mathrm{Lip}_1(\mathbb{R}^D)} \frac{1}{n} \sum_{i=1}^n f(\hat{X}_i) - f(X_i) \\
&\leqslant \mathbb{E} \sup_{f \in \mathrm{Lip}_1(\mathbb{R}^D)} \frac{1}{n} \sum_{i=1}^n |f(\hat{X}_i) - f(X_i)| = \mathbb{E} \sup_{f \in \mathrm{Lip}_1(\mathbb{R}^D)} \frac{1}{n} \sum_{i=1}^n |f(X_i + \xi_i) - f(X_i)| \\
&\leqslant \mathbb{E} \sup_{f \in \mathrm{Lip}_1(\mathbb{R}^D)} \frac{1}{n} \sum_{i=1}^n \|\xi_i\|_2 = \mathbb{E}\|\xi\|_2 \leqslant \sqrt{V_\xi}
\end{aligned}$$

the last line follows from Jensen's inequality. □

We conclude in the noisy setting that

$$\mathbb{E}W_1((\hat{g}_n)_\sharp \rho, Q) \leqslant \epsilon_{\mathrm{appx}} + 2C_\delta n^{-\frac{1}{d+\delta}} + 2\sqrt{V} \leqslant C n^{-\frac{1}{d+\delta}} + 2\sqrt{V_\xi}$$

after balancing the approximation error ϵ_{appx} appropriately.