On Noise Injection in Generative Adversarial Networks

Ruili Feng
University of Science and Technology of China.
ruilifengustc@gmail.com

Deli Zhao
Xiaomi AI Lab.
zhaodeli@gmail.com

Zhengjun Zha
University of Science and Technology of China.
zhazj@ustc.edu.cn

Abstract

Noise injection has been proved to be one of the key technique advances in generating high-fidelity images. Despite its successful usage in GANs, the mechanism of its validity is still unclear. In this paper, we propose a geometric framework to theoretically analyze the role of noise injection in GANs. Based on Riemannian geometry, we successfully model the noise injection framework as fuzzy equivalence on geodesic normal coordinates. Guided by our theories, we find that existing methods are incomplete and a new strategy for noise injection is devised. Experiments on image generation and GAN inversion demonstrate the superiority of our method.

1 Introduction

Noise injection has attracted more and more attention in the community of Generative Adversarial Networks (GANs) [10]. Extensive research works show that it helps generate images of high fidelity and variety [14, 8], stabilizes the training procedure [2, 13], and enhances the robustness of neural networks [22, 4, 11]. Traditional GANs take in randomness only from input latent codes and the GAN frameworks with noise injection seek to inject extra noise directly to intermediate feature spaces of the network, as illustrated in Figure 1. It is interesting to point out that this modification of the generator coincides with the gene expression process, where information from a gene is used in the synthesis of a functional gene product (like proteins). It is well known that genotypes along with environmental factors together determine what the phenotypes will be [20, 25, 31]. From a cognitive aspect, twins share the same genes yet admit subtle differences in appearance. So the gene expression could be modeled as a random output network, where genes control the mean and variance of observable traits, while the environment injects the randomness [9, 29, 6]. These designs allows genes, with relatively simple structures, to express highly complicated, fine, diverse biological phenotypes. If comparing input latents in GANs with genes, then random noise injection plays the role of environmental influences. This bionic principle contributes to the bio-inspired motivation of noise injection, as real world data also contain subtle and abundant details, e.g. textures, wrinkles, and patterns of surroundings, thus posing a challenge for the relatively low-dimensional latents to fit. In practice, extensive experiments in StyleGAN [14] and BigGAN [5] also confirm its validity.

In this work, we propose a theoretic framework to explain and improve the effectiveness of noise injection. Our framework is motivated from a geometric perspective yet also combined with the results of optimal transportation problem in GANs [18, 19]. Our contributions are as follows:
We show that the existing GAN architectures, including the Wasserstein GANs [3], may suffer from adversarial dimension trap, which severely penalizes the property of generator;

Based on our theory, we try to explain the properties that noise injection is applied in the related literatures;

Based on our theory, we find a more proper form for noise injection in GANs, which can overcome the adversarial dimension trap, and experiments on the state-of-the-art GAN architecture, StyleGAN2 [15], demonstrate the superiority of our new method compared with original noise injection used in StyleGAN2.

To the best of our knowledge, this is the first work that theoretically draws the geometric picture of noise injection in GANs.

2 Related works

The main drawbacks of GANs are unstable training and mode collapse. Arjovsky et al. [2] theoretically analyze that noise injection directly to the image space can help smooth the distribution so as to stabilize the training procedure. The authors of Distribution-Filtering GAN (DFGAN) [13] then put this idea into practice and prove that this technique will not influence the global optimality of the real data distribution. However, as the authors pointed out in [2], this method depends on the amount of noise, and the Jenson-Shannon divergence is not an intrinsic measure of synthesis and data distributions. Actually, our method of noise injection is essentially different from these ones. Besides, they do not provide a theoretical vision of the interactions between injected noises and the features.

BigGAN [5] splits input latents into one chunk per layer as injected noise and project each chunk to the gains and biases of batch normalization in each layer. They claim that this design allows direct influence on features at different resolutions and levels of hierarchy. StyleGAN [14] and StyleGAN2 [15] adopt a slightly different view, where noise injection is introduced to enhance randomness for multi-scale stochastic variations. Different from the settings in BigGAN, they inject extra noise independent of the latent inputs into different layers of the network without projection but a learnable scalar scale. Our proposed framework reveals noise injection in StyleGAN as fuzzy reparameterization in Euclidean spaces, and extend it into generic manifolds (section 4.3).
3 The intrinsic drawbacks of traditional GANs

3.1 Optimal transportation and discontinuous generator

Traditional GANs with Wasserstein distance are equivalent to the optimal transportation problem, where the optimal generator is the optimal transportation map. However, there is rare chance for the optimal transportation map to be continuous, unless the support of Brenier potential is convex [7]. Considering that the Brenier potential of Wasserstein GAN is determined by the real data distribution and the inverse map of the generator, it is highly unlikely that its support is convex. This means that the optimal generator will be discontinuous, which is a fatal limitation to the capacity of GANs. Based on that, Lei et al. [18] further point out that, traditional GANs will hardly converge or converge to one continuous branch of the target mapping, leading to mode collapse. They then propose to find the continuous Brenier potential instead of the discontinuous transportation map. While in the next paragraph, we show that this solution may not totally overcome the problem that the traditional GANs encounter, given the structural limitations of neural networks. We refer the readers to [18, 7] for more detailed demonstration.

3.2 Adversarial dimension trap

Besides the above discontinuity problem, another drawback is the relatively low dimension of latent spaces compared with the high variance of the details in real world data. Picking images of human faces as an example, the hair, freckles, and wrinkles have extremely high degree of freedom, which make traditional GANs often fail to capture them. The reiterative application of non-invertible CNN blocks make the situation even worse. Non-invertible CNN, which is a singular linear transformation, will drop the intrinsic dimensions of feature manifolds as information passed [28]. So during the feedforward procedure of the generator, dimensions of the feature spaces will keep dropping. Then, it will have a high chance that the valid dimension of the input latent space is lower than that of the real data. The relatively lower dimension of the input latent space will then force the dimension of the support of the distribution of generated images lower than that of the real data, as no smooth mappings could increase the dimension. However, the discriminator, which measures the distance of these two distributions, will keep encouraging the generator to increase the dimension up to the same as the true data’s. This contradictory functionality, as we show in the theorem bellow, induces severe punishment on the smoothness and invertibility of the generative model, which we refer as the adversarial dimension trap.

**Theorem 1. [1]** For a deterministic GAN model and generator \( G : \mathcal{Z} \to \mathcal{X} \), if the dimension of \( \mathcal{Z} \) is lower than \( \mathcal{X} \)'s, then at least one of the two cases must stand:

1. the generator cannot be Lipschitz;
2. the generator fails to capture the data distribution and is unable to perform inversion.

Notice that this theorem implies much worse situation than it states. As for any open sphere \( B \) in the data manifold \( \mathcal{X} \), the generator restricted in the pre-image of \( B \) also follows this theorem, which suggests bad properties of nearly every local neighborhood.

The first issue can be addressed by not learning the generator directly by continuous neural network components, which suffices our proposed method in the next section. We will show how our method addresses the second issue.

4 Fuzzy reparameterization

The generator \( G \) in the traditional GAN is a composite of a series non-linear feature mappings, which can be denoted as \( G(z) = f^k \circ f^{k-1} \circ \cdots \circ f^1(z) \), where \( z \sim \mathcal{N}(0, I) \) is the standard Gaussian. Each feature mapping, which is typically a single layer CNN plus non-linear activations, carries out a certain purpose such as extracting multi-scale patterns, upsampling, or merging multi-head information. The whole network is then a deterministic mapping from the latent space \( \mathcal{Z} \) to the image.
We call it as a fuzzy reparameterization (FR) as it in fact learns a fuzzy equivalence relation of the original features, and uses reparameterization to model the high dimensional feature manifolds. We believe that this is the proper form of generalization of noise injections in GANs, and will show the reasons and benefits in the following sub-sections.

It is not hard to see that our proposed method can be viewed as the extension of the reparameterization trick in VAEs [16]. While the reparameterization trick in VAEs serves to a differentiable solution to random variables and is only applied in the latent space, our method is a type of deep noise injection on feature maps of each layer to fix the defects in GAN architectures. Therefore, the purposes of using reparameterization in these two scenarios are different, thus leading to different theories that are presented in the next sub-section.

4.1 Handling adversarial dimension trap through noise injection

As Sard’s theorem tells us [22], the key to solve the adversarial dimension trap is to avoid mapping low-dimensional feature spaces into high-dimensional ones, which looks like a pyramid structure. However, we really need the pyramid structure of the generator because the final output dimension of generated images is much larger than that of the input latents. So the solution could be that, instead of mapping into the full feature spaces, we choose to map only onto the skeleton of the feature spaces and use random noises to fill up the remaining space. For a compact manifold, it is easy to find that the intrinsic dimension of the skeleton set can be arbitrarily low by applying Heine–Borel theorem to the skeleton [26]. By this way, the model can escape from the adversarial dimension trap.

Now we develop the idea in detail. The whole idea is based on approximating the manifold by the tangent polyhedron. Assume that the feature space $\mathcal{M}$ is Riemannian manifold embedded in $\mathbb{R}^m$. Then for any point $\mu \in \mathcal{M}$, the local geometry induces a coordinate transformation from a small neighborhood of $\mu$ in $\mathcal{M}$ to its projection onto the tangent space $T_\mu \mathcal{M}$ at $\mu$ by the following theorem.

**Theorem 2.** Given Riemannian manifold $\mathcal{M}$ embedded in $\mathbb{R}^m$, for any point $\mu \in \mathcal{M}$, let $T_\mu \mathcal{M}$ be the tangent space at $\mu$, then the exponential map $\text{Exp}_\mu$ induces a smooth diffeomorphism from a Euclidean ball $B_{T_\mu \mathcal{M}}(0, r)$ centered at $O$ to a geodesic ball $B_{\mathcal{M}}(\mu, r)$ centered at $\mu$ in $\mathcal{M}$. Thus $\{\text{Exp}_\mu^{-1}, B_{\mathcal{M}}(\mu, r), B_{T_\mu \mathcal{M}}(0, r)\}$ forms a local coordinate system of $\mathcal{M}$ in $B_{\mathcal{M}}(\mu, r)$, which we call the normal coordinates. Thus we have:

$$B_{\mathcal{M}}(\mu, r) = \text{Exp}_\mu(B_{T_\mu \mathcal{M}}(0, r)) = \{\tau : \tau = \text{Exp}_\mu(v), v \in B_{T_\mu \mathcal{M}}(0, r)\}. \quad (2)$$

**Theorem 3.** The differential of $\text{Exp}_\mu$ at the origin of $T_\mu \mathcal{M}$ is identity $I$. Thus $\text{Exp}_\mu$ can be approximated by

$$\text{Exp}_\mu(v) = \mu + Iv + o(\|v\|_2). \quad (3)$$

Thus, if $r$ in equation (2) is small enough, we can approximate $B_{\mathcal{M}}(\mu, r)$ by:

$$B_{\mathcal{M}}(\mu, r) \approx \mu + IB_{T_\mu \mathcal{M}}(0, r) = \{\tau : \tau = \mu + Iv, v \in B_{T_\mu \mathcal{M}}(0, r)\}. \quad (4)$$
Considering that $T_{\mu}M$ is an affine subspace of $\mathbb{R}^m$, the coordinates on $B_{T_{\mu}M}(0, r)$ admit an affine transformation into the coordinates on $\mathbb{R}^m$. Thus equation (4) can be written as:

$$B_{M}(\mu, r) \approx \mu + IB_{T_{\mu}M}(0, r) = \{ \tau : \tau = \mu + rT(\mu)\epsilon, \epsilon \in B(0, 1) \}. \tag{5}$$

We remind the readers that the linear component matrix $T(\mu)$ differs at different $\mu \in M$ and is decided by the local geometrics near $\mu$.

In this formula, $\mu$ defines the center point and $rT(\mu)$ defines the shape of the approximated neighbor. So we call them a representative pair of $B_{M}(\mu, r)$. Picking up a series of such representative pairs, which we refer as the skeleton set, we can construct a tangent polyhedron $\mathcal{H}$ of $M$. Thus instead of trying to learn the feature manifold directly, we adopt a two-stage procedure. We first learn a map $f : x \mapsto [\mu(x), \sigma(x)] (\sigma(x) \equiv rT(\mu(x)))$ onto the skeleton set, then we use noise injection $g : x \mapsto \mu(x) + \sigma(x)\epsilon, \epsilon \sim \mathcal{U}(0, 1)$ (uniform distribution) to fill up the flesh of the feature space as shown in Figure 2.

However, the real-world data often include fuzzy semantics. Even spatially-remote features could share some relations in common. It might be unwise to model it with unsmooth architectures such as local bounded sphere and uniform distribution. Thus we borrow the idea from fuzzy topology \[20, 33, 21, 24\] which is designed to address this issue. It is well known that for any distance metrics $d(\cdot, \cdot), e^{-d(\cdot, \cdot)}$ admits a fuzzy equivalence relation for points near $\mu$, which is similar with the density of Gaussian. The fuzzy equivalence relation could be viewed as a suitable smooth alternative to the sphere neighbor $B_{M}(\mu, r)$. Thus we replace the uniform distribution with unclipped Gaussian \[\frac{\sigma}{2}\] Under this settings, the first stage mapping in fact learns a fuzzy equivalence relation, while the second stage is a reparameterization technique. Notice that the skeleton set can have arbitrarily low dimension by Heine–Borel theorem. So the first-stage map can be smooth and well conditioned. For the second stage, we can show that it possess a smooth property in expectation by the following theorem.

**Theorem 4.** Given $f : x \mapsto [\mu(x), \sigma(x)]^T$, $f$ is locally Lipschitz and $\|\sigma\|_{\infty} = o(1)$. Define $g(x) \equiv \mu(x) + \sigma(x)\epsilon, \epsilon \sim \mathcal{N}(0, 1)$ (standard Gaussian), then for any bounded set $U$, s.t. $\mathbb{E}(|g(x) - g(y)|_2 \leq L \|x - y\|_2 + o(1), \forall x, y \in U$. Namely, the principal component of $g$ is locally Lipschitz in expectation. Specifically, if the definition domain of $f$ is bounded, then the principal component of $g$ is globally Lipschitz in expectation.

### 4.2 Properties of noise injection

As we have discussed, traditional GANs face two challenges: the discontinuous optimal generator and the adversarial dimension trap. Both of the two challenges will lead to an unsmooth generator. And Theorem 1 also implies an unstable training procedure considering the gradient explosion that may occur on the generator. Besides, the dimension drop in GAN will cause it hard to fit high-variance details as information keeps compressed along channels in the generator. While we apply noise injection to the network, we can theoretically overcome such problems if the representative pairs are constructed properly to capture the local geometrics. In that case, our model do not need to fit the discontinuous optimal transportation map, nor the image manifold with higher dimension than the network architecture can hold. Thus the training procedure will not encourage unsmooth generator, and can proceed more stably. Also, the extra noise can fill the loss of information compression so as to capture high-variance details, which has been discussed and proved in StyleGAN [14]. We will experimentally evaluate the performance of our method from these aspects in section 5.

### 4.3 Choice of $\mu(x)$ and $\sigma(x)$

As $\mu(x)$ stands for a particular point in the feature space, we simply model it by the traditional deep CNN architectures. $\sigma(x)$ is designed to fit the local geometrics of $\mu(x)$. While we assume the feature space to be smooth manifold, the local geometrics should only admit minor differences from $\mu(x)$. Thus we believe that $\sigma(x)$ should be determined by the spatial and semantic information contained in

---

\[\frac{\sigma}{2}\] A detailed analysis about why unclipped Gaussian should be applied based on fuzzy topology is offered in the supplementary material.
Table 1: Metrics for various generator architectures. For StyleGAN2-based models, PPL scores are measured in the intermediate latent spaces.

| GAN arch                  | FFHQ     | LSUN-Church |       |       |       |       |
|---------------------------|----------|-------------|-------|-------|-------|-------|
|                            | PPL (↓)  | FID (↓)     | IS (↑) | Precision (↑) | PPL (↓) | FID (↓) | IS (↑) | Precision (↑) |
| DCGAN                     | 2.97     | 45.29       | 2.42  | 0.764 | 33.30 | 51.18  | 2.56   | 0.646 |
| DCGAN + Additive noise    | 3.14     | 44.22       | 2.52  | 0.761 | 22.97 | 54.01  | 2.68   | 0.477 |
| DCGAN + FR                | 2.83     | 40.86       | 2.53  | 0.767 | 22.53 | 46.31  | 2.56   | 0.582 |
| Bald StyleGAN2            | 28.44    | 6.87        | 4.54  | 0.673 | 425.7 | 6.44   | 2.58   | 0.573 |
| StyleGAN2-NoPathReg       | 19.69    | 6.38        | 4.57  | 0.670 | 322.0 | 5.88   | 2.62   | 0.567 |
| StyleGAN2                 | 16.20    | 7.29        | 4.62  | 0.682 | 119.5 | 6.86   | 2.51   | 0.589 |
| StyleGAN2-NoPathReg + FR  | 16.02    | 7.14        | 4.61  | 0.666 | 178.9 | 5.75   | 2.62   | 0.567 |
| StyleGAN2 + FR            | 13.05    | 7.31        | 4.58  | 0.681 | 119.5 | 6.86   | 2.49   | 0.600 |

\( \mu(x) \). This vision then guides us into the following settings:

\[
\mu = \text{DCNN}(x), \quad \sigma = \text{PixSum}(\mu),
\]

\[
\sigma = \alpha(A \ast (\sigma - \text{mean}(\sigma))/\max(|\sigma|) + b) + (1 - \alpha)I, \quad \alpha \in [0, 1],
\]

\[
\sigma = r\sigma/\|\sigma\|_2, \quad r > 0, \quad \mu = \sigma \ast \mu, \quad o = \mu + \sigma \ast \epsilon, \quad \epsilon \sim \mathcal{N}(0, 1),
\]

where PixSum stands for pixel-wise sum, \( o \) is the output, and \( A, b, \alpha, r \) are learnable parameters. There are alternative forms for \( \mu \) and \( \sigma \) with respect to various GAN architectures. However, modeling \( \mu \) by deep CNNs and deriving \( \sigma \) through the spatial and semantic information of \( \mu \) are universal for GANs, as they suit our theorem.

StyleGAN is also equipped with noise injection. But different from our design, they adopt a simple solution as follows:

\[
\mu = \text{DCNN}(x), \quad o = \mu + r \ast \epsilon, \quad \epsilon \sim \mathcal{N}(0, 1),
\]

where \( r \) is a learnable \textit{scalar} parameter. This can be viewed as a special case of our method, where \( T(\mu) \) in (5) is set to identity. Under this settings, the local geometrics are assumed to be universal among the feature manifold, which suggests a globally Euclidean structure. While our theory supports this simplification and specialization, our choice of \( \mu(x) \) and \( \sigma(x) \) can suit broader and more usual occasions, where the feature manifolds are non-Euclidean. We denote this fashion of noise injection as additive noise injection, and will extensively study its performance compared with our choice in the following section.

5 Experiment

We conduct experiments on FFHQ faces, LSUN objects, and CIFAR-10 multiple classes. The GAN models we use are the baseline DCGAN [23] (originally with no noise injection) and the state-of-the-art StyleGAN2 [15] (originally with additive noise injection). For StyleGAN2, we use config-e in the original paper due to that config-e achieves the best performance of Path Perceptual Length (PPL) score. We inherit all the other experimental settings from StyleGAN2.

Image synthesis. The PPL has been proven an effective metric for the perceived image quality. Considering its similarity to the expectation of the Lipschitz constant of the generator, it can also be viewed as a quantification of the smoothness of the generator. The path length regularizer is proposed in StyleGAN2 to regularize the PPL score by explicitly regularizing the Jacobian of the generator with respect to the intermediate latent space. We first compare the noise injection methods with the bald StyleGAN2, which remove the additive noise injection and path length regularizer in StyleGAN2. Through all the cases we can find that all types of noise injection significantly improve the PPL scores. As it is shown in Table 1, our method, without path length regularizer, can achieve comparable performance against StyleGAN2 in FFHQ dataset, and the performance can be further improved if combined with path length regularizer. Considering the extra GPU memory consuming of path length regularizer in training procedure, we believe that our method offers a computation-friendly alternative to StyleGAN2 as we observe smaller GPU memory usage of our method throughout all the experiments. Another benefit is that our method accelerates the convergence to the optimal FID scores. Figure 4 reports the FID curve as the iteration evolves. By our theorem, this can be explained...
as our method offers an architecture that is more consistent to the intrinsic geometrics of the feature space. Thus it is easier for the network to fit.

In the LSUN-Church dataset, we observe an obvious improvement in FID scores compared with StyleGAN2. We believe that, this is because the the LSUN-Church data are scene images and contain various semantics of multiple objects, which are hard to fit for the original StyleGAN2 that is more suitable for single object synthesis. So our FR architecture offers more degree of freedom to the generator to fit the true distribution of the dataset. In all cases, our method outperforms StyleGAN2-
NoPathReg, which removes the path length regularizer in StyleGAN2, and is superior to StyleGAN2 in PPL scores if our method is combined with path length regularizer. This proves that our noise injection method is more proper then the one used in StyleGAN2. For DCGAN, as it does not possess the intermediate latent space, we cannot facilitate it with path length regularizer. So we only compare the additive noise injection with our FR method. Through all the cases we can find that our method achieves the best performance in PPL and FID scores.

We also want to study whether our choice for $\mu(x)$ and $\sigma(x)$ can be applied to broader occasions. We further conduct experiments on a cat dataset, which consists of 100 thousand selected images from 800 thousand LSUN-Cat images by PageRank algorithm, to test whether our method could succeed with insufficient training images. For DCGAN, we conduct extra experiments on CIFAR-10, to test whether our method could succeed in multi-class image synthesis. The results are reported in Figure 5, where we can see our method still dominates the performance against the compared methods in PPL scores with comparable FIDs.

Numerical stability. As we have analyzed before, noise injection should be able to improve the numerical stability of GAN models. To evaluate it, we examine the condition number of different GAN architectures. The definition of the condition number of a given function $f$ can be found at [12]. It measures how sensitive a function is to changes or errors in the input, and how many errors in the output results from an error in the input. Considering the numerical infeasibility of the sup operator in the definition of condition number for functions, we resort to the following alternatives. We sample a batch of 50000 pairs of $(Input, Perturbation)$ severally from the input distribution and $\mathcal{N}(0, 1e-4)$, and compute the corresponding condition numbers. We compute the mean value and the mean value of the largest 1000 values of these 50000 condition numbers as Mean Condition (MC) and Top Thousand Mean Condition (TTMC) correspondingly to evaluate the condition of GAN models. We report the results in Table 2 where we can find that noise injection significantly improves the condition of GAN models, and our propose method dominates the performance.
GAN inversion. StyleGAN2 makes use of an intermediate latent space that holds the promise of enabling some controlled image modifications. This characteristic motivates us to study the image embedding capability of our method via GAN inversion algorithms [1] as it may help further leverage the potential of GAN models. During the experiments, we find that the StyleGAN2 model develops an appetite for full-face, non-blocking human face images. For this type of images, we observe comparable performances for all the GAN architectures. We believe that this is because those images are close to the ‘mean’ face of FFHQ dataset [14], thus easy to learn for the StyleGAN-based models. For faces of large pose or partially blocked ones, the capacity of different models differs significantly. Noise injection methods outperform the bald StyleGAN2 by a large margin, and our method achieves the best performance. The detailed implementation and results are reported in the supplementary material.

6 Conclusion

In this paper, we propose a theoretical framework to explain the effect of noise injection technique in GANs. We prove that the generator can easily become unsmooth, and noise injection is a useful tool against it. We also derive a more proper formulation for noise injection based on our theoretical framework. We conduct experiments on various datasets to confirm its validity. Despite of the superiority compared with the existing methods, however, it is still unclear whether it is optimal and universal among different networks and datasets for the specific neural network implementation of our formulation. In future work, we will further investigate the detailed implementation of noise injection, and find out much more powerful way to model local geometrics of feature spaces.

Appendices

A Proof to theorems

A.1 Theorem 1

Proof. Denote the dimensions of $G(Z)$ and $X$ as $d_G$ and $d_X$, respectively. There are two possible cases for $G$: $d_G$ is lower than $d_X$, or $d_G$ is higher than or equal to $d_X$.

For the first case, a direct consequence is that, for almost all points in $X$, there are no pre-images under $G$. This means that for an arbitrary point $x \in X$, the possibility of $G^{-1}(x) = \emptyset$ is 1, as $\{x \in X : G^{-1}(x) \neq \emptyset\} \subset G(Z) \cap X$, which is a zero measure set in $X$. That means that the generator is unable to perform inversion. Another consequence is that, the generated distribution $P_G$ can never get aligned with real data distribution $P_r$. Namely, the distance between $P_r$ and $P_G$ cannot be zero for whatever distance metric. For the KL divergence, the distance will even become infinity.

For the second case, where $d_G \geq d_X > d_Z$. We simply show that a Lipschitz continuous function cannot map zero measure set into positive measure set. Specifically, the image of low dimensional space of a Lipschitz continuous function has measure zero. Thus if $d_G \geq d_X$, $G$ cannot be Lipschitz.

Now we prove our claim.

Suppose $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, $n < m$, $f$ is Lipschitz with Lipschitz constant $L$. We show that $f(\mathbb{R}^n)$ has measure zero in $\mathbb{R}^m$. As $\mathbb{R}^n$ is a zero measure subset of $\mathbb{R}^m$, by the Kirszbraun theorem [8], $f$ has an extension, still denoted as $f$, to a Lipschitz function of the same Lipschitz constant on $\mathbb{R}^m$. Then the problem reduces to proving that $f$ maps zero measure set to zero measure set. For every $\epsilon > 0$, we can find countable union of balls $\{B_k\}_k$ of radius $r_k$ such that $\mathbb{R}^n \subset \bigcup_k B_k$ and $\sum_k m(B_k) < \epsilon$ in $\mathbb{R}^m$, where $m(\cdot)$ is the Lebesgue measure in $\mathbb{R}^m$. But $f(B_k)$ is contained in a ball with radius $Lr_k$. Thus we have $m(f(\mathbb{R}^n)) \leq L^m \sum_k m(B_k) < L^m \epsilon$, which means that it is a zero measure set in $\mathbb{R}^m$. For the mappings between manifolds, using the chart system can turn it into the cases we analyze above, which completes our proof.

We also want to remind the readers that, even if the generator suits one of the cases in Theorem 1, the other case can still occur. For example, $G$ could succeed in capturing the distribution of certain parts.
of the real data, while fail in the other parts. Then for the pre-image of those successfully captured data, the generator will not have finite Lipschitz constant.

A.2 Theorems 2 & 3

Theorems 2 & 3 are classical conclusions in Riemannian manifold. We refer readers to section 5.5 of [22] for detailed proofs and illustration.

A.3 Theorem 4

Proof.

\[ E[\|g(x) - g(y)\|_2] \leq \|\mu(x) - \mu(y)\|_2 + E[\|\sigma(x)\epsilon - \sigma(y)\delta\|_2] \]
\[ \leq L_\mu\|x - y\|_2 + 2C\|\sigma\|_\infty \leq L_\mu\|x - y\|_2 + o(1), \]

where \( C \) is a constant related to the dimension of the image space of \( \sigma \) and \( L_\mu \) is the Lipschitz constant of \( \mu \).

B Why Gaussian distribution should be applied

We first introduce the notion of fuzzy equivalence relations.

Definition 1. A t-norm is a function \( T: [0, 1] \times [0, 1] \rightarrow [0, 1] \) which satisfies the following properties:

1. Commutativity: \( T(a, b) = T(b, a) \).
2. Monotonicity: \( T(a, b) \leq T(c, d) \), if \( a \leq c \) and \( b \leq d \).
3. Associativity: \( T(a, T(b, c)) = T(T(a, b), c) \).
4. The number 1 acts as identity element: \( T(a, 1) = a \).

Definition 2. Given a t-norm \( T \), a \( T \)-equivalence relation on a set \( X \) is a fuzzy relation \( E \) on \( X \) and satisfies the following conditions:

1. \( E(x, x) = 1, \forall x \in X \) (Reflexivity).
2. \( E(x, y) = E(y, x), \forall x, y \in X \) (Symmetry).
3. \( T(E(x, y), E(y, z)) \leq E(x, z) \forall x, y, z \in X \) (T-transitivity).

Then it is easy to check that \( T(x, y) = xy \) is a t-norm, and \( E(x, y) = e^{-d(x,y)} \) is a \( T \)-equivalence for any distance metric \( d \) on \( X \), as

\[ T(E(x, y), E(y, z)) = e^{-d(x,y)+d(y,z)} \leq e^{-d(x,z)} = E(x, z). \]

Considering that we want to contain the fuzzy semantics of real world data in our local geometrics of feature manifolds, a natural solution will be that, we sample points from the local neighbor of \( \mu \) with different density on behalf of different strength of semantic relations with \( \mu \). Those who contain stronger semantic relations would have larger densities to be sampled. A good framework to model this process is the fuzzy equivalence relations we mentioned above, where the degrees of membership \( E \) is used as the sampling density. However, remember that our expansion of the exponential map \( \text{Exp}_\mu \) carries an error term of \( o(\|v\|_2) \). We certainly do not want the local error to be out of control, and we also wish to constrain the sampling locally. Thus we accelerate the decrease of density when points depart from the center \( \mu \), and constrain the integral of \( E \) to be identity, which turns \( E \) into the density of standard Gaussian.

C Datasets

FFHQ  Flickr-Faces-HQ (FFHQ) [14] is a high-quality image dataset of human faces, originally created as a benchmark for generative adversarial networks (GANs). The dataset consists of 70,000 high-quality PNG images and contains considerable variations in terms of age, pose, expression, hair style, ethnicity and image backgrounds. It also has good coverage of accessories such as eyeglasses, sunglasses, hats, etc.
Table 3: GPU environments for all experiments in this work.

| Experiment                        | Environment                                      |
|-----------------------------------|--------------------------------------------------|
| StyleGAN2 based GAN model training| 8 NVIDIA Tesla V100-SXM2-16GB GPUs (DGX-1 station) |
| DCGAN based GAN model training    | 4 TITAN Xp GPUs                                   |
| Metrics measurement              | 8 GeForce GTX 1080Ti GPUs                         |
| GAN inversion                     | 1 TITAN Xp GPU                                    |

**LSUN-Church and Cat-Selected**  LSUN-Church is the church outdoor category of LSUN dataset [32], which consists of 126 thousands church images of various styles. Cat-Selected are 100 thousands cat images selected by ranking algorithm [35] from the cat category.

**CIFAR-10**  The CIFAR-10 dataset [17] consists of 60,000 images of size 32x32. There are all 10 classes and 6000 images per class. There are 50,000 training images and 10000 test images.

### D Implementation details

#### D.1 Metrics

**Mean Condition (MC) and Top Thousand Mean Condition (TTMC)**  The condition number of a given function $f$ is defined as

$$\text{Cond}(f) = \lim_{\delta \to 0} \sup_{\|\Delta x\| \leq \delta} \frac{\| f(x) - f(x + \Delta x) \| / \| f(x) \|}{\| \Delta x \| / \| x \|}.$$  (12)

It measures how sensitive a function is to changes or errors in the input, and how many errors in the output results from an error in the input. A function with a low condition number is said to be well-conditioned, while a function with a high condition number is said to be ill-conditioned. Considering the numerical infeasibility of the $\sup$ operator in the definition of condition number for functions, we resort to the following alternatives. We sample a batch of 50000 pairs of $(x, \Delta x)$ from the input distribution and $N(0, 1e-4)$, and compute the corresponding condition numbers. We compute the mean value and the mean value of the largest 1000 values of these 50000 condition numbers as Mean Condition (MC) and Top Thousand Mean Condition (TTMC) respectively to evaluate the condition of GAN models.

**Other metrics**  For all the other metrics, we inherit the implementation and settings from the official StyleGAN2 code.

#### D.2 Models

We illustrate the generator architectures of StyleGAN2 based methods in Figure 6. For all those models, the discriminators share the same architecture as the original StyleGAN2. The generator architecture of DCGAN based methods are illustrated in Figure 7. For all those models, the discriminators share the same architecture as the original DCGAN.

### E Experiment environment

All experiments are carried out by TensorFlow 1.14 and Python 3.6 with CUDA Version 10.2 and NVIDIA-SMI 440.64.00. We basically build our code upon the framework of NVIDIA official StyleGAN2 code, which is available at https://github.com/NVlabs/stylegan2 We use a variety of servers to run the experiments as reported in Table 3.

### F Image encoding and GAN inversion

StyleGAN2 makes use of an intermediate latent space that holds the promise of enabling some controlled image modifications. This characteristic motivates us to study the image embedding capability of our method as it may help further leverage the potential of GAN models. Also, from a
mathematical perspective, a well behaved generator should be easily invertible. In the last section, we have found that our method is well conditioned, which implies that it could be easily invertible. We adopt the methods in Image2StyleGAN [1] to perform GAN inversion and compare the mean square error and perceptual loss on a manually collected dataset of 20 images. The images are shown in Figure 8 and the quantitative results are provided in Table 4. For our FR methods, we further optimize the $\alpha$ parameter in Eq. 7 of section 4.3, which finetunes the local geometrics of the network to suit the new images that might not be settled in the model. Considering that $\alpha$ is limited to $[0, 1]$, we use...
(a) DCGAN. (b) Additive Noise Injection. (c) Fuzzy Reparameterization.

Figure 7: Generator architecture of DCGAN based models. (a) The generator of DCGAN. (b) The generator of DCGAN + Additive Noise. (c) The generator of DCGAN + FR.

Table 4: Image inversion metrics for different StyleGAN2 based models. The perceptual loss is the mean square distance of VGG16 features between the original and projected images as in [1].

| GAN arch | Overall | Hard Cases |
|----------|---------|------------|
|          | MSE ↓ | Perceptual Loss ↓ | MSE ↓ | Perceptual Loss ↓ |
| Bald StyleGAN2 | 1.34 | 5.42 | 2.86 | 11.34 |
| StyleGAN2-NoPathReg | 1.26 | 4.70 | 2.65 | 9.29 |
| StyleGAN2 | 1.24 | 4.86 | 2.58 | 9.82 |
| StyleGAN2-NoPathReg + FR | 1.24 | 5.11 | 2.70 | 10.49 |
| StyleGAN2 + FR | **1.13** | **4.52** | **2.23** | **8.47** |

\[
\frac{(\alpha^*)t}{(\alpha^*)t + (1 - \alpha^*)r} \text{ to replace the original } \alpha \text{ and optimize } t. \text{ The initial value of } t \text{ is set to 1.0 and } \alpha^* \text{ is constant with the same value as } \alpha \text{ in the converged FR models.}
\]

During the experiments, we find that the StyleGAN2 model develops an appetite for full-face, non-blocking human face images. For this type of images (which we refer as regular case in Figure 9), we observe comparable performance for all the GAN architectures. We believe that this is because those images are closed to the ‘mean’ face of FFHQ dataset [13], thus easy to learn for the StyleGAN based models. For faces of large pose or partially blocked ones, the capacity of different models differs significantly. Noise injection methods outperform the bald StyleGAN2 by a large margin, and our method achieves the best performance.

References

[1] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2StyleGAN: How to embed images into the StyleGAN latent space? In Proceedings of the IEEE International Conference on Computer Vision, pages 4432–4441, 2019.

[2] Martin Arjovsky and Léon Bottou. Towards principled methods for training generative adversarial networks. arxiv e-prints, art. arXiv preprint arXiv:1701.04862, 2017.
Figure 8: Manually collected 20 images for GAN inversion.

| Origin | Bald StyleGAN2 | StyleGAN2-NoPathReg | StyleGAN2 | StyleGAN2-NoPathReg + FR |
|--------|----------------|----------------------|-----------|--------------------------|

Regular case

Hard case

Figure 9: Projected Images to the intermediate latent space of StyleGAN2 based models.

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. arXiv preprint arXiv:1701.07875, 2017.

[4] Chaim Baskin, Natan Liss, Yoav Chai, Evgenii Zheltonozhskii, Eli Schwartz, Raja Giryes, Avi Mendelson, and Alexander M Bronstein. Nice: Noise injection and clamping estimation for neural network quantization. arXiv preprint arXiv:1810.00162, 2018.

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large-scale GAN training for high fidelity natural image synthesis, 2018.

[6] Daniela Bustos-Korts, Marcos Malosetti, Scott Chapman, and Fred van Eeuwijk. Modelling of genotype by environment interaction and prediction of complex traits across multiple environ-
ments as a synthesis of crop growth modelling, genetics and statistics. In *Crop systems biology*, pages 55–82. Springer, 2016.

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

[8] Klaus Deimling. *Nonlinear functional analysis*. Courier Corporation, 2010.

[9] Jean-Baptiste Denis, Hans-Peter Piepho, and Fred A van Eeuwijk. Modelling expectation and variance for genotype by environment data. *Heredity*, 79(2):162–171, 1997.

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

[11] Zhezhi He, Adnan Siraj Rakin, and Deliang Fan. Parametric noise injection: Trainable randomness to improve deep neural network robustness against adversarial attack. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 588–597, 2019.

[12] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 2013.

[13] Simon Jenni and Paolo Favaro. On stabilizing generative adversarial training with noise. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12145–12153, 2019.

[14] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[15] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. *arXiv preprint arXiv:1912.04958*, 2019.

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

[17] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[18] Na Lei, Yang Guo, Dongsheng An, Xin Qi, Zhongxuan Luo, Shing-Tung Yau, and Xianfeng Gu. Mode collapse and regularity of optimal transport maps. *arXiv preprint arXiv:1902.02934*, 2019.

[19] Na Lei, Kehua Su, Li Cui, Shing-Tung Yau, and Xianfeng David Gu. A geometric view of optimal transportation and generative model. *Computer Aided Geometric Design*, 68:1–21, 2019.

[20] Zhang Ling and Zhang Bo. Theory of fuzzy quotient space (methods of fuzzy granular computing). 2003.

[21] V Murali. Fuzzy equivalence relations. *Fuzzy sets and systems*, 30(2):155–163, 1989.

[22] Peter Petersen, S Axler, and KA Ribet. *Riemannian geometry*, volume 171. Springer, 2006.

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

[24] Jordi Recasens. *Indistinguishability operators: Modelling fuzzy equalities and fuzzy equivalence relations*, volume 260. Springer Science & Business Media, 2010.

[25] I Romagosa and PN Fox. Genotype × environment interaction and adaptation. In *Plant breeding*, pages 373–390. Springer, 1993.

[26] Walter Rudin et al. *Principles of mathematical analysis*, volume 3. McGraw-hill New York, 1964.

[27] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[28] Gilbert Strang, Gilbert Strang, Gilbert Strang, and Gilbert Strang. *Introduction to linear algebra*, volume 3. Wellesley-Cambridge Press Wellesley, MA, 1993.

[29] Fred A van Eeuwijk. *Between and beyond additivity and non-additivity: the statistical modelling of genotype by environment interaction in plant breeding*. Van Eeuwijk, 1996.
[30] Sara Via and Russell Lande. Genotype-environment interaction and the evolution of phenotypic plasticity. *Evolution*, 39(3):505–522, 1985.

[31] Brian Westcott. Some methods of analysing genotype—environment interaction. *Heredity*, 56(2):243–253, 1986.

[32] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

[33] Ling Zhang and Bo Zhang. Fuzzy reasoning model under quotient space structure. *Information Sciences*, 173(4):353–364, 2005.

[34] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.

[35] Dengyong Zhou, Jason Weston, Arthur Gretton, Olivier Bousquet, and Bernhard Schölkopf. Ranking on data manifolds. In *Advances in neural information processing systems*, pages 169–176, 2004.