

# *Generative Adversarial Network Performance in Low-Dimensional Settings*

Felix Jimenez<sup>1,2</sup>, Amanda Koepke<sup>1</sup>, Mary Gregg<sup>1</sup>, and Michael Frey<sup>1</sup>

<sup>1</sup>National Institute of Standards and Technology,  
Gaithersburg, MD 20899, USA

<sup>2</sup>University of Colorado Boulder,  
Boulder, CO, 80309, USA

[felix.jimenez@nist.gov](mailto:felix.jimenez@nist.gov)  
[amanda.koepke@nist.gov](mailto:amanda.koepke@nist.gov)  
[mary.gregg@nist.gov](mailto:mary.gregg@nist.gov)  
[michael.frey@nist.gov](mailto:michael.frey@nist.gov)

A generative adversarial network (GAN) is an artificial neural network with a distinctive training architecture, designed to create examples that faithfully reproduce a target distribution. GANs have recently had particular success in applications involving high-dimensional distributions in areas such as image processing. Little work has been reported for low dimensions, where properties of GANs may be better identified and understood. We studied GAN performance in simulated low-dimensional settings, allowing us to transparently assess effects of target distribution complexity and training data sample size on GAN performance in a simple experiment. This experiment revealed two important forms of GAN error, tail underfilling and bridge bias, where the latter is analogous to the tunneling observed in high-dimensional GANs.

**Key words:** earth mover distance; experiment protocol; generative adversarial network; mode tunneling; modeling error; target distribution complexity.

**Accepted:** March 12, 2021

**Published:** April 20, 2021

<https://doi.org/10.6028/jres.126.008>

---

## 1. Introduction

A generative network is an artificial neural network (ANN) designed to synthesize examples similar in desired ways to the examples on which the network is trained. The examples presented to the generative network during training are understood to be points in a sample  $S$  drawn from a target distribution  $T$ , and the generative network seeks to learn  $T$ . Because the learning process is imperfect, the target distribution  $T$  and the distribution  $L$  actually learned by the generative network will differ, and this error—the distance  $d(T, L)$  between the two distributions—determines the generative network’s performance. An example of generative modeling is the task of creating faces for cartoon characters [1]. A generative network is trained with examples of cartoon faces taken from available hand-drawn animations so that it can synthesize faces that are original but yet, for example, have the desired hand-drawn appearance and adherence to face conventions (nose above mouth, etc.). The target distribution  $T$  in this example is the population of all cartoon faces with the desired characteristics. Based on a sample  $S$  of training faces from  $T$ , the generative network learns to synthesize faces with distribution  $L$ , ideally with an error  $d(T, L)$  so small that a face randomly generated from  $L$  and a target face from  $T$  are indistinguishable in the desired ways.

Generative adversarial networks (GANs), introduced by Goodfellow *et al.* [2] and Creswell *et al.* [3], are a form of generative network with a novel training architecture in which the generative network, termed the generator, is trained alongside a classifier network, termed the discriminator. In this arrangement the generator and discriminator are pitted one against the other and trained simultaneously in iterative fashion: At each iteration the generator presents learned examples to the discriminator, and the discriminator attempts to correctly identify these learned examples from among a pool of training examples. The discriminator reports its successes and failures back to the generator, each network updates itself based on the discriminator's successes and failures, and a new iteration begins. At each iteration the generator is trying to mislead the discriminator, and the discriminator is trying to avoid being misled. After enough iterations, the generator and discriminator approach a game-theoretic equilibrium where the discriminator cannot distinguish between synthesized examples and training examples any better than guessing [2, 3]. At this equilibrium the generator—the GAN—is ready for use.

GANs have been successfully applied in many fields; for example, drug design [4], galactic astronomy [5], and health care [6]. GANs have been applied most prominently and extensively to image processing, for image generation and image-to-image translation [2, 7, 8], to generate fake celebrity faces [9], for super-resolution [10], to give images the style of another image [11], and to selectively alter images [12]. These applications all involve inputs and outputs that are both high-dimensional features.<sup>1</sup> Generative networks are distinctive in this respect; the outputs of ANNs designed for classification or prediction are commonly just one-dimensional features, a categorical variable for ANN classifiers and a real-valued variable for ANN predictors. The high-dimensional problem settings in which GANs are applied complicate study of GAN performance because of the multitude of effects associated with the rich detail in the input during training and the challenge of assessing GAN performance from its high-dimensional distributional output. Additionally, the high dimensionality of both input and output imposes a computational burden on experiments with GANs. Directly addressing GAN performance in high-dimensional application settings has yielded only limited progress on pressing issues such as mode collapse [13], model evaluation [14], and training instability due to saddlepoints [15, 16], hidden low-dimensional target support [17], and absence of an equilibrium [18].

To avoid the complications entailed by high-dimensional inputs and outputs, we took a different tack and studied GANs in low-dimensional settings. This enabled us to transparently address some basic questions about GAN performance. In the work described in this paper, we conducted an experiment to explore the effects of training sample size and target distribution complexity on GAN error. Target distribution complexity is determined in significant part by the distribution's number of modes. For example, consider the 28×28-pixel images of the 10 digits in the Modified National Institute of Standards and Technology (MNIST) data set [19]. The MNIST target distribution has nominally 10 modes within its 28×28-dimensional support, and its complexity can be varied by excluding digits from consideration. Modes—their placements, sizes, and feature associations—are not readily interpreted or manipulated in many high-dimensional applications, where mode shape<sup>2</sup> is a complicating factor. In low-dimensional settings, by contrast, multimodal target distributions are easily rendered, for example, by mixtures of Gaussian distributions.

This paper reveals important features of GANs and sets a path for further study. We focused on GAN performance in low-dimensional settings, where experiments can be conducted with a manageable computational burden. This approach also allows the experimenter to control for confounding effects and achieve transparent results. The experiment conducted in this study introduces and verifies procedures (for

---

<sup>1</sup>In image processing the dimensionality of the GAN input is of the order of the image size in pixels.

<sup>2</sup>The shape of a mode can be intuited in terms of the shape and dimensionality of the submanifold on which it resides; a mode in one dimension is a zero-dimensional point, and a mode in two dimensions might be a point or a section of a curve (on a one-dimensional sub-manifold).

GAN error measurement, design, stability, and training convergence) that can be used in subsequent experiments with broader scope and more factors. Also, this study emphasizes the primacy of the relationship of GAN error to the size  $N$  of the training sample, and our experiment finds that GAN error is log-log linear with respect to  $N$ . Surprisingly, in the cases in our experiment, this relationship extends to even small  $N$ , allowing GAN error to be understood succinctly in terms of its error exponent (log-log slope). Finally, our experiment uncovers two forms of GAN error, which we term tail underfilling and bridge bias. This demonstrates that GAN mode tunneling (of which bridge bias is the low-dimensional analogue) occurs, and can be studied, in low-dimensional GANs. This is a validation of our choice to study GAN performance in low-dimensional settings.

The remainder of this paper is organized as follows. Section 2 presents important details of our experiment, including the experiment protocol followed and our use of earth mover (EM) distance to quantify GAN error  $d(L, T)$ . We also present the architecture of the GAN in our experiment and our GAN training procedure. Section 3 presents and discusses the results of our experiment. Section 4 closes with a summary and some related final remarks.

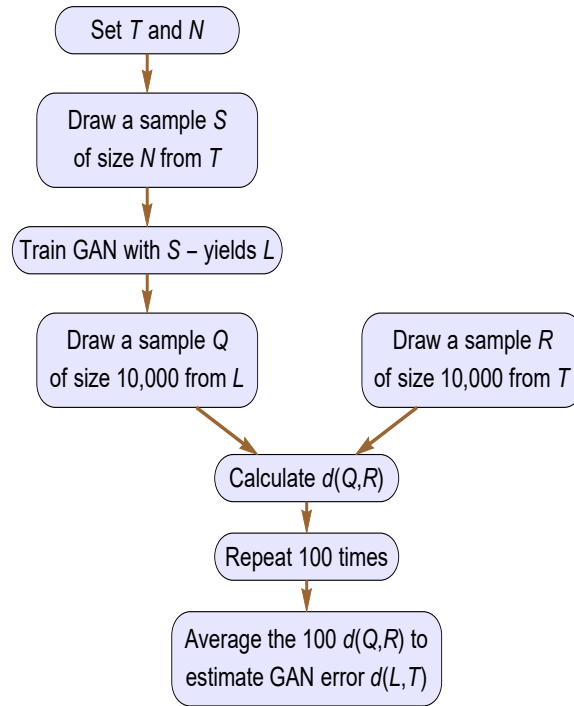
## 2. Preliminaries

This section details the way our experiment was conducted. This includes, in particular, the protocol used for the experiment's trials and the particular form of GAN studied. We also discuss our use of EM distance to quantify GAN error.

### 2.1 Protocol for Experiment Trials

The primary objective of our study was to determine how training sample size affects GAN performance for different degrees of target distribution complexity. Our experiment to explore this relationship among GAN error, sample size, and target complexity consisted of trials in which our GAN was trained and exercised over a range of training sample sizes  $N$  for each of six target distributions  $T$  with varying complexity, including three one-dimensional distributions and three two-dimensional distributions.

All trials in our experiment were conducted according to the protocol diagrammed in Fig. 1. Each trial began with a given target distribution  $T$  and a size  $N$  for the training sample  $S$ . The training sample  $S$  was drawn by simple random sampling from  $T$  and used to train the GAN. Once trained, the GAN generator synthesized a sample  $Q$  of size 10,000 from the GAN's learned distribution  $L$ , a second sample  $R$  of size 10,000 was drawn from  $T$ , and the distance  $d(Q, R)$  was calculated. This distance  $d(Q, R)$  is an approximation of the GAN error  $d(L, T)$ . This whole process was repeated 100 times (a new GAN trained each time), and the GAN error  $d(L, T)$  for the trial was estimated by the average  $\bar{d}(Q, R)$  of the 100  $d(Q, R)$ .



**Fig. 1.** Protocol for trials in GAN performance experiment. The GAN learns  $L$  from a training sample  $S$ . The GAN error  $d(L, T)$  is then estimated by comparing samples  $Q$  and  $R$  from  $L$  and  $T$ , respectively.

## 2.2 GAN Fundamentals

A GAN has two components in its training phase, its generator,  $G_\theta$ , and a discriminator,  $D_\phi$ , where  $\theta$  and  $\phi$  are vectors of parameters estimated in the learning process. The generator  $G_\theta$  is a mapping from a noise (latent) distribution  $Z$  to its learned distribution  $L$ . The discriminator  $D_\phi$  assigns a probability<sup>3</sup> that a presented example, whether from  $T$  or synthesized, comes from  $T$ . Both  $G_\theta$  and  $D_\phi$  are typically implemented by ANNs. These two networks are trained according to a combined loss function [21]

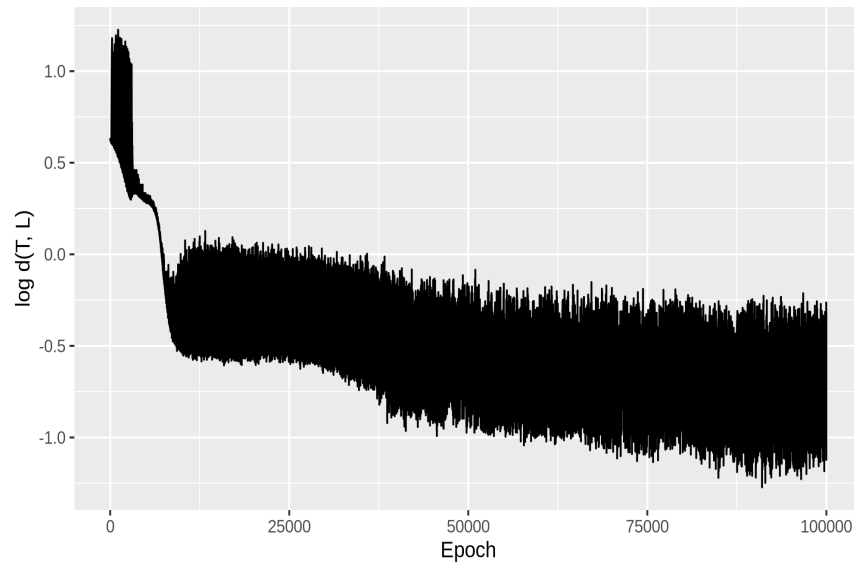
$$\mathcal{L}(\theta, \phi) = E_{\zeta \sim Z}[f(D_\phi(G_\theta(\zeta)))] + E_{\tau \sim T}[f(-D_\phi(\tau))]. \quad (1)$$

At each iteration in training,  $G_\theta$  tries to minimize  $\mathcal{L}(\theta, \phi)$  while  $D_\phi$  tries to maximize  $\mathcal{L}(\theta, \phi)$ . Popular GAN variants are distinguished by different choices of  $f(x)$  in Eq. (1). For example, the original Goodfellow GAN [2] and the Wasserstein GAN [20] correspond, respectively, to  $f(x) = -\ln(1 + e^{-x})$  and  $f(x) = x$ . When the GAN discriminator is optimal, the GAN generator minimizes the distance between the target distribution  $T$  and the distribution  $L$  learned by the generator; the Goodfellow GAN minimizes the Jensen-Shannon divergence, and the Wasserstein GAN minimizes the EM distance between  $L$  and  $T$ . We used in our experiment a Wasserstein GAN with a gradient penalty term, called a WGAN-GP [22], added to the loss in Eq. (1). Table 1 summarizes the architecture of our WGAN-GP; this architecture is recommended in Ref. [24]. The generator output dimension is either one or two depending on the dimension of the target distribution in our experiment. We used an Adam optimizer [27] for both the generator and discriminator with a learning rate of  $1e-4$ , with  $\beta_1 = 0.5$  and  $\beta_2 = 0.9$ . For each update of the generator, we performed

<sup>3</sup>A discriminator that outputs a class membership probability rather than just a class label is sometimes termed a critic [20].

five updates of the discriminator. We found these values to give relatively consistent, stable results during training. Choosing training parameters remains an art based on experience and trial-and-error.

The GAN training process is notoriously unstable, and various expediencies have been proposed in the literature to stabilize training [15]. These include changes to the loss function used in training [20, 22], tricks for improving GAN training [15, 23], and investigations of GAN convergence [24, 26]. Our WGAN-GP is known to exhibit training instabilities [24], but we encountered none in our low-dimensional setting. We found with the WGAN-GP that the GAN error  $d(L, T)$  consistently reached steady-state after training for 50,000 epochs.<sup>4</sup> Figure 2 shows the evolution of GAN error with number of learning epochs for a typical trial in our experiment.



**Fig. 2.** GAN error versus number of epochs for a representative experiment trial. In our experiment, the GAN was trained in each trial for 50,000 epochs.

**Table 1.** Architecture of GAN generator (left) and discriminator (right). The generator input is a vector of independent  $N(0, 1)$  noises with a length of 128. The leaky-relu function is defined  $f(x) = 1(x < 0)(\alpha x) + 1(x \geq 0)(x)$ , where  $\alpha$  is a small constant.

Generator				Discriminator			
Layer	Connectivity	No. of units	Activation	Layer	Connectivity	No. of units	Activation
1	Full	128	Leaky-relu	1	Full	32	Leaky-relu
2	Full	128	Leaky-relu	2	Full	32	Leaky-relu
3	Full	128	Leaky-relu	3	Full	32	Leaky-relu
4	Full	Dim. of $T$	None	4	Full	1	Leaky-relu

<sup>4</sup>An epoch is one complete use of the training sample  $S$  during learning. Usually, only a subset (batch) of  $S$  is used in any given learning iteration, so one epoch typically corresponds to multiple iterations.

### 2.3 EM Distance

Our study of GAN performance needed a measure of distance between distributions to quantify GAN error. EM distance is attractive for this purpose, and hereafter,  $d(U, V)$  denotes the EM distance between distributions  $U$  and  $V$ . EM distance is a special case ( $p = 1$ ) of  $p$ -Wasserstein distance, which measures separation among probability distributions and is a metric in the general setting of Radon spaces [25]. Let  $U$  and  $V$  be two distributions on  $d$ -dimensional Euclidean space, where  $\mathcal{R}^d$  is defined by cumulative distribution functions (CDFs)  $F$  and  $G$ , respectively. The EM distance separating the distributions  $U$  and  $V$  is

$$d(U, V) = \inf_H \int_{\mathcal{R}^d \times \mathcal{R}^d} \|\vec{u} - \vec{v}\| dH(\vec{u}, \vec{v}), \quad (2)$$

where  $H$  is any joint CDF on  $\mathcal{R}^d \times \mathcal{R}^d$  such that, marginally,  $U \sim F$  and  $V \sim G$ . EM distance can be understood intuitively as the infimum cost required to move/rearrange a probability mass distributed according to  $U \sim F$  into the distribution of  $V \sim G$ , with Euclidean distance  $\|\cdot\|$  measuring the move required for each infinitesimal of probability mass. The joint CDF  $H$  in the infimum in Eq. (2) represents different possible plans for transporting each infinitesimal of probability mass from  $U$  to  $V$ .

EM distance has a long history, stretching back to Monge's 1781 work in transportation theory [28]. Recently, EM distance has been used broadly in computer science, with applications to pattern recognition [29], image databasing [30], and content-based image retrieval [31]. Arjovsky *et al.* used EM distance to formulate the Wasserstein GAN [20] to address issues of mode collapse and vanishing gradients, two problems inherent in the original GAN framework [2] and of continuing concern. EM distance is just one of many distance measures that can be defined for probability distributions [32], and no consensus has yet emerged for measuring GAN error [33]. Among measures of distance between distributions, EM distance has a powerful and prevailing role in many fields because of its sensitivity to both amount of mass and to underlying metrical, or ground, distance. This feature of EM distance, its dual sensitivity to mass and distance, makes it attractive for our purposes as a direct measure of GAN error.

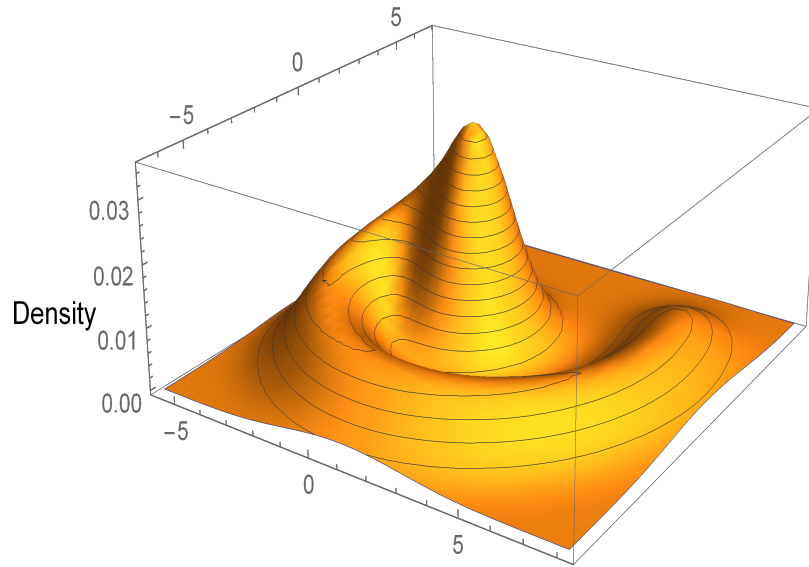
EM distance can be expressed analytically in some limited cases, and in one dimension the sample  $S$  closest in EM distance to  $T$  takes a simple form. In general, though, EM distance is found numerically using Sinkhorn's algorithm to solve a regularized version of the basic optimal transport problem [34, 35]. We used the Python Optimal Transport Library [36] implementation of Sinkhorn's algorithm in this work to calculate GAN error  $d(L, T)$ .

## 3. Experiment Results

This section presents and discusses the results of our target distribution complexity experiment in which the relationship of GAN error to training sample size was studied as it varied with the complexity of the target distribution. The experiment included six target distributions, three one-dimensional distributions and three two-dimensional distributions. The three one-dimensional target distributions had one, two, and three modes. As remarked earlier, adding more modes can be thought of as adding more digits in the MNIST data set. The three two-dimensional distributions in this experiment were a Gaussian distribution, an equal mixture of two Gaussian distributions, and a Swiss roll distribution. The Swiss roll distribution, commonly used in machine learning studies [37], is given by the random vector

$$(X_1, X_2) = (U \cos U, U \sin U) + (W_1, W_2),$$

where  $W_1$ ,  $W_2$ , and  $U$  are independent random variates with  $U \sim \text{Unif}(0, 9\pi/4)$  and  $W_1, W_2 \sim \text{N}(0, 0.01)$ . The density of the Swiss roll distribution is shown in Fig. 3.



**Fig. 3.** Density of the Swiss roll target distribution.

### 3.1 Results

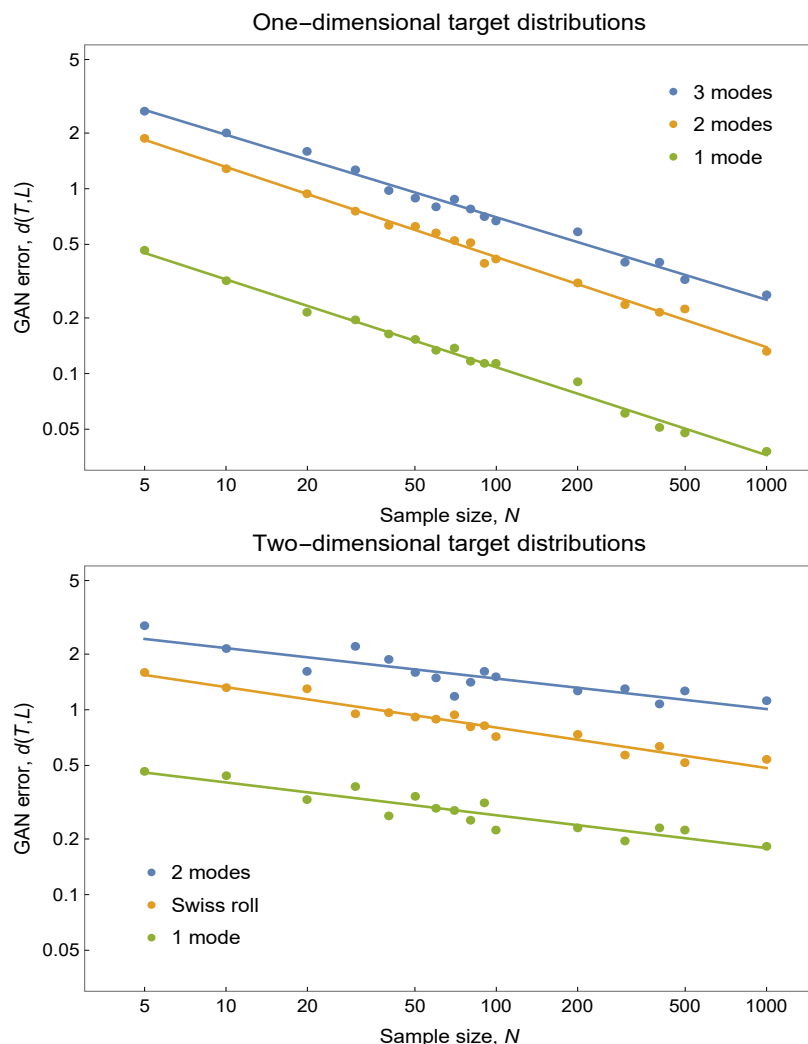
The results from the target distribution complexity experiment are plotted in Fig. 4. First, these results confirm that increased target complexity results in poorer GAN performance and greater error. We see this in Fig. 4 for the one-dimensional  $T$  in the ordering of higher GAN error with greater number of modes. Figure 4 shows this also for the two-dimensional  $T$  in the experiment, provided the Swiss roll  $T$  is interpreted as more complex than a distribution with a single point mode but less complex than a  $T$  with two point modes. Second, in the log-log format of Fig. 4, the relationship of GAN error  $d(L, T)$  to sample size  $N$  appears to be linear

$$\log d(L, T) = a - b \log N \quad (3)$$

or, equivalently,  $d(L, T) = 10^a N^{-b}$ , with error exponent  $b$ . This linear relationship might be anticipated to hold asymptotically for large  $N$ ; in the results in Fig. 4, it appears to apply even at small sample sizes. Analysis of a standard analysis of covariance (ANCOVA) model confirms this. That analysis indicates further that the error exponent  $b$  in Eq. (3) differs significantly according to the dimensionality of  $T$ , but that no statistically discernible differences exist (at  $\alpha = 0.05$ , general linear  $F$  test [38]) among the error exponents for the one-dimensional  $T$  or among those for the two-dimensional  $T$  distributions. The common error exponents are estimated to be 0.47 and 0.19, respectively, for the one- and two-dimensional  $T$  distributions in the experiment.

The GAN errors  $d(L, T)$  shown in Fig. 4 were obtained according to the protocol in Fig. 1. The protocol's size, using 10,000 samplings of  $Q$  and  $R$  and its numerical approximation of  $d(Q, R)$  together impose a noise floor on the GAN error that can be resolved by this approach. This floor can be determined by using the protocol to find  $d(L, T)$  in the extreme case where the GAN learns the target distribution perfectly so that  $L = T$  with zero GAN error. These protocol-estimated  $d(T, T)$  floors are reported in Table 2 for each target distribution in our experiment. GAN errors below these floors cannot be accurately determined by our protocol with size-10,000  $Q$  and  $R$  samples; larger sample sizes would lower these floors and increase the protocol's resolution. The floors in Table 2 show that  $Q$  and  $R$  samples of size 10,000 are sufficient for our experiment.





**Fig. 4.** GAN error  $d(L, T)$  for one-dimensional target distributions with one, two, and three modes (top) and for three two-dimensional target distributions (bottom). For the one-dimensional  $T$ , the GAN error decreases with sample size  $N$  at a common rate with estimated error exponent 0.47. For two-dimensional  $T$ , the error decreases at a common rate with estimated error exponent 0.19.

The measurements of GAN error  $d(L, T)$  in Fig. 4 made according to the protocol in Fig. 1 were produced on the Raritan computing cluster at the National Institute of Standards and Technology. Twelve of the cluster's Nvidia Volta graphics processing units (GPUs) were used in parallel for the experiment<sup>5</sup>. Training a GAN to 50,000 epochs (Fig. 2) took about 45 min. By training 12 GANs in parallel, each GAN error  $d(L, T)$  in the experiment was calculated in 5 to 8 h. The six error curves in Fig. 4, each estimated from 16 error measurements, therefore required a total of about 600 h (25 d) to complete.

<sup>5</sup>Certain commercial equipment, instruments, or materials are identified in this paper in order to specify the experimental procedure adequately. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

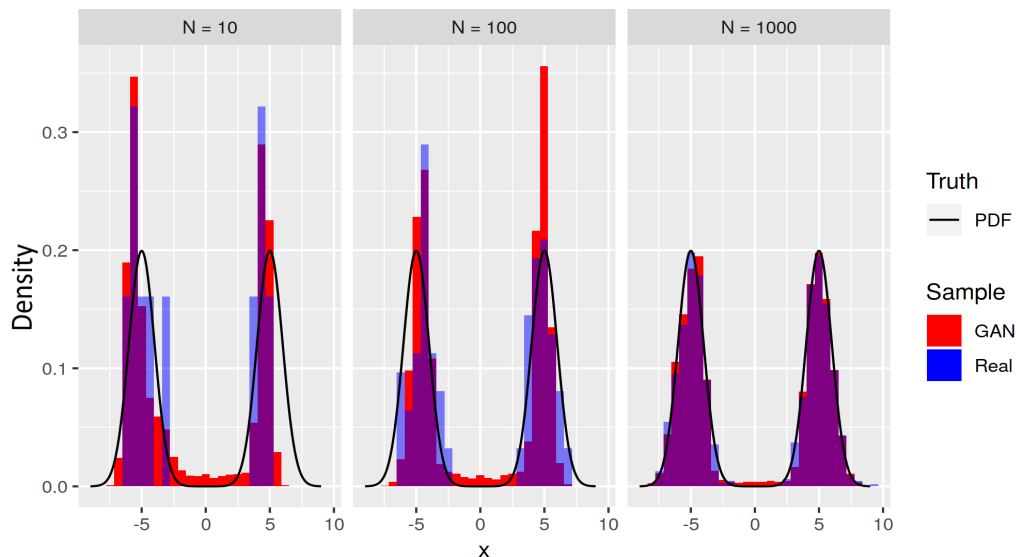


**Table 2.** Measurement floors for GAN error for the six target distributions in Fig. 4. These floors were estimated by  $d(T, T)$  with size 10,000 samples from  $T$ .

Target distribution $T$	GAN error measurement floor
One-dimensional distributions:	
Unimodal Gaussian	0.018
Bimodal Gaussian mixture	0.072
Trimodal Gaussian mixture	0.117
Two-dimensional distributions:	
Unimodal Gaussian	0.057
Swiss roll	0.080
Bimodal Gaussian mixture	0.042

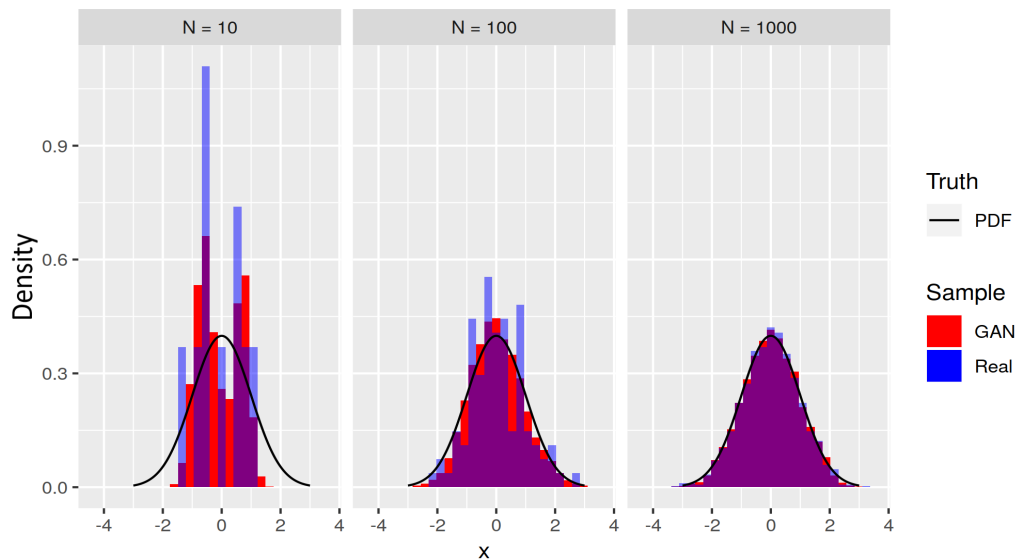
### 3.2 Discussion

The low-dimensional setting of our experiment readily revealed two major sources for the GAN errors presented in Fig. 4: underfilled tails and bridge bias. Figure 5 presents three trials with a target standard normal  $N(0, 1)$  distribution, using training sample sizes  $N$  ranging from 10 to 1000. Fig. 5 shows that at small training sample sizes  $N$ , the GAN's learned distribution, represented by the examples in red, underfills the tails of the target distribution (in black), and this error diminishes as  $N$  increases. Figure 5 also suggests that this is not inherently a problem with the GAN. The GAN can do no better than the training data available to it, and the poor fidelity of the training sample (in blue) at small  $N$  is at least partially the origin of the underfilled tails.



**Fig. 5.** Histograms (red) of learned distributions  $L$  for a bimodal target distribution  $T$  (black), for three sizes  $N = 10$ , 100, and 1000 of training sample (blue). Each histogram of examples from  $L$  exhibits a bridge bias where the GAN has generated examples inconsistent with both the density of the bridge in  $T$  and the training data drawn from distribution  $T$ .

Figure 6 shows results of a GAN trained on a bimodal target distribution (in black) made of an equal mixture of two normal distributions  $N(-5, 1)$  and  $N(5, 1)$ . The target distribution  $T$  has a low-density bridge connecting its two modes. Figure 6 shows that for small  $N$  the GAN's learned distribution (examples in red) over-estimates the bridge density, even though the training sample (in blue) actually underrepresents the target bridge. The side experiment described next suggests that this bridge bias is a GAN structural feature that is only slowly corrected by increasing the size  $N$  of the training sample.

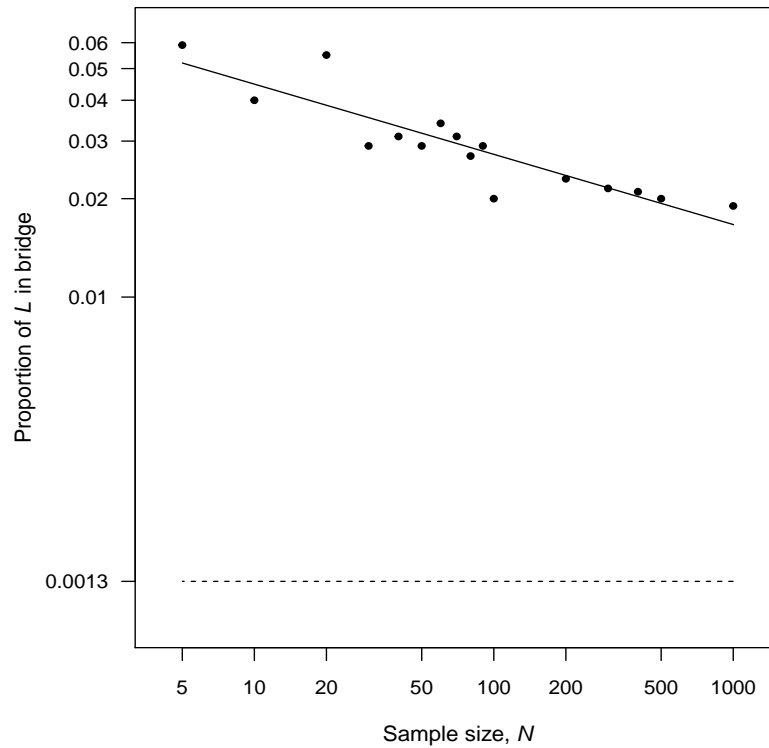


**Fig. 6.** Histograms (red) of learned examples from distribution  $L$  for a unimodal target distribution  $T$  (black) for three sizes  $N = 10, 100$ , and  $1000$  of training sample (blue). The learned distribution  $L$  underfills the tails of  $T$  at small sample sizes  $N$ .

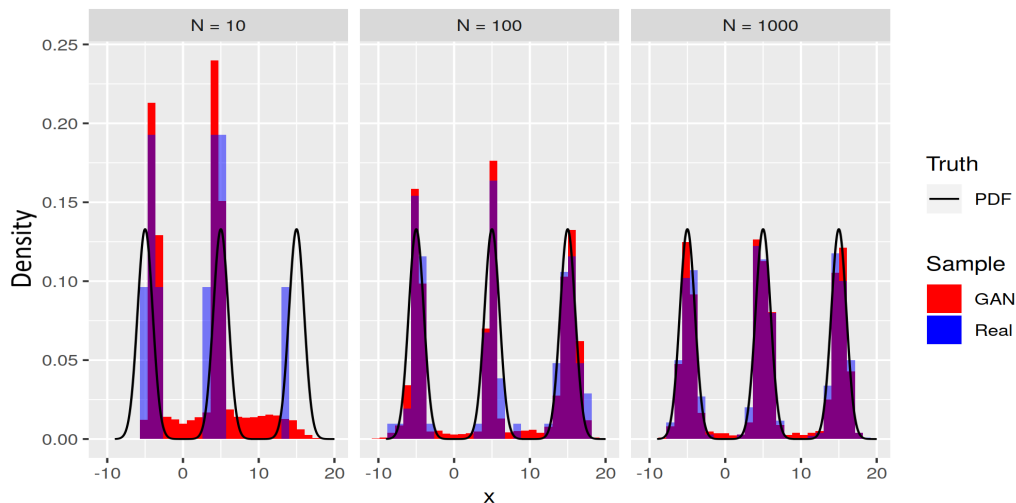
A side experiment separate from our main target complexity experiment was conducted to explore the bridge bias seen in Fig. 6. This side experiment looked at only the univariate bimodal target distribution  $T$  of Fig. 6, estimating the size of the bridge in the GAN-learned distribution  $L$  over a set of training sample sizes  $N$ . The results of this side experiment are presented in Fig. 7, where the proportion of  $L$  in  $T$ 's bridge<sup>6</sup> is plotted against  $N$ . These results show diminishing bridge bias as the training sample size increases. However, extrapolation of the regression line (solid line) in Fig. 7 suggests that very large sample sizes would be needed—even in this simple one-dimensional learning problem—to approach the target bridge proportion (dashed line) and effectively eliminate the bridge bias in  $L$ .

The trimodal target distribution in Fig. 8 shows that for a given sample size  $N$ , bridge bias worsens in the presence of multiple target bridges. Figures 6 and 8 show the presence of GAN bridge bias in examples where the target distribution modes are connected by bridges. In fact, GAN bridge bias occurs even when no bridge connects different modes of the target distribution, as demonstrated when the training sample happens to have no examples between two modes.

<sup>6</sup>The target distribution  $T$  is a bimodal mixture of Gaussian distributions  $N(-5, 1)$  and  $N(5, 1)$ . We defined  $T$ 's bridge to be that part of  $T$  that falls more than three standard deviations below its upper mode and more than three standard deviations above its lower mode. With this definition the bridge constitutes 0.0013 of  $T$ .

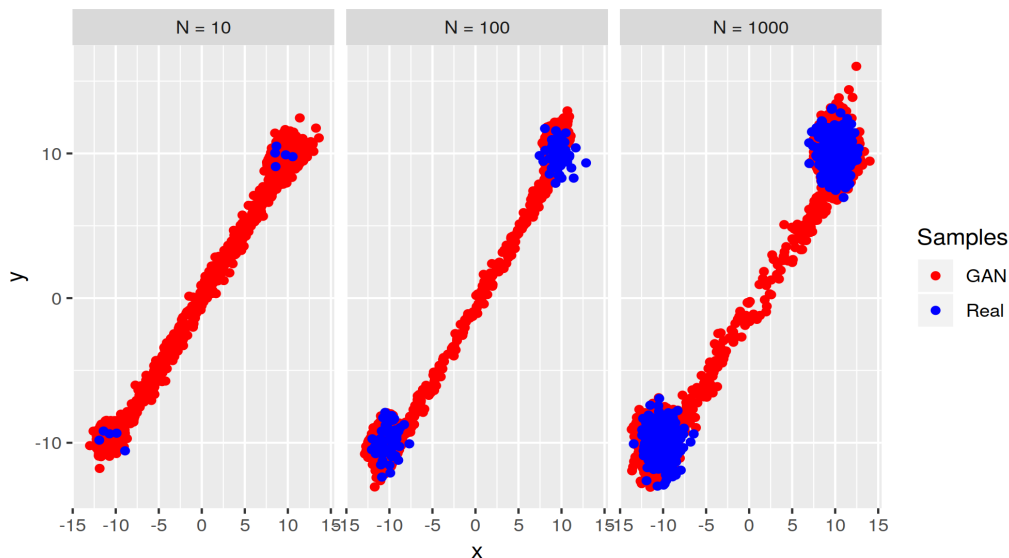


**Fig. 7.** Bridge bias in the GAN-learned distribution  $L$  of the univariate bimodal target distribution  $T$ . The plotted proportions of  $L$  in the bridge of  $T$  decrease slowly with increasing training sample size  $N$ . The learned distribution  $L$  over-estimates the proportion of  $T$  in its bridge (dashed line).



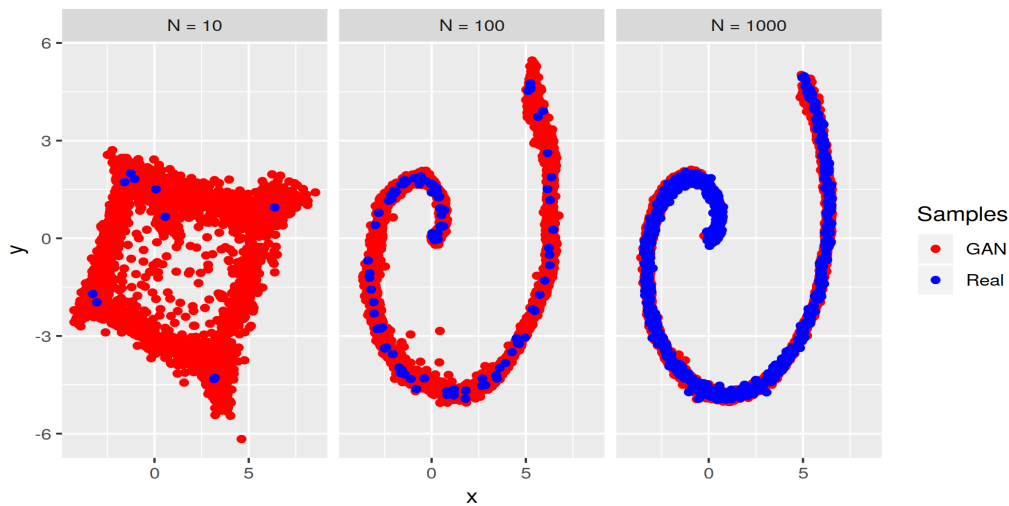
**Fig. 8.** Histograms (red) of learned distributions  $L$  for a trimodal target distribution  $T$  (black), for three sizes  $N = 10$ , 100, and 1000 of training sample (blue). Each histogram of examples from  $L$  exhibits a bridge bias where the GAN has generated examples inconsistent with both the density of the bridge in  $T$  and the training data drawn from distribution  $T$ .

The GAN bridge bias seen with our one-dimensional multimodal target distributions appears also with our two-dimensional distribution  $T$ . Figure 9 shows GAN-generated samples for the target bimodal mixture of bivariate Gaussian distributions in our experiment. In each of the three cases in Fig. 9— $N = 10$ , 100, and 1000—the learned bridge between the two modes greatly over-estimates the (very low) density of the bridge in the target distribution. This bias is greater than that for our one-dimensional target distributions. Also, while the bridge bias for two-dimensional  $T$  decreases with  $N$ , this decrease is slower than that in the corresponding one-dimensional  $T$ . Interestingly, this bridge bias in two dimensions is restricted mostly to the one-dimensional line between the two modes, meaning that while the bridge bias for two-dimensional  $T$  is more pronounced compared to that for one-dimensional  $T$ , it is also more restricted relative to the support of  $T$ . Experiments to explore the relationships among amount and extent of GAN bridge bias, GAN architecture, training parameters, training sample size, and target complexity (dimensionality and multimodality) are needed.



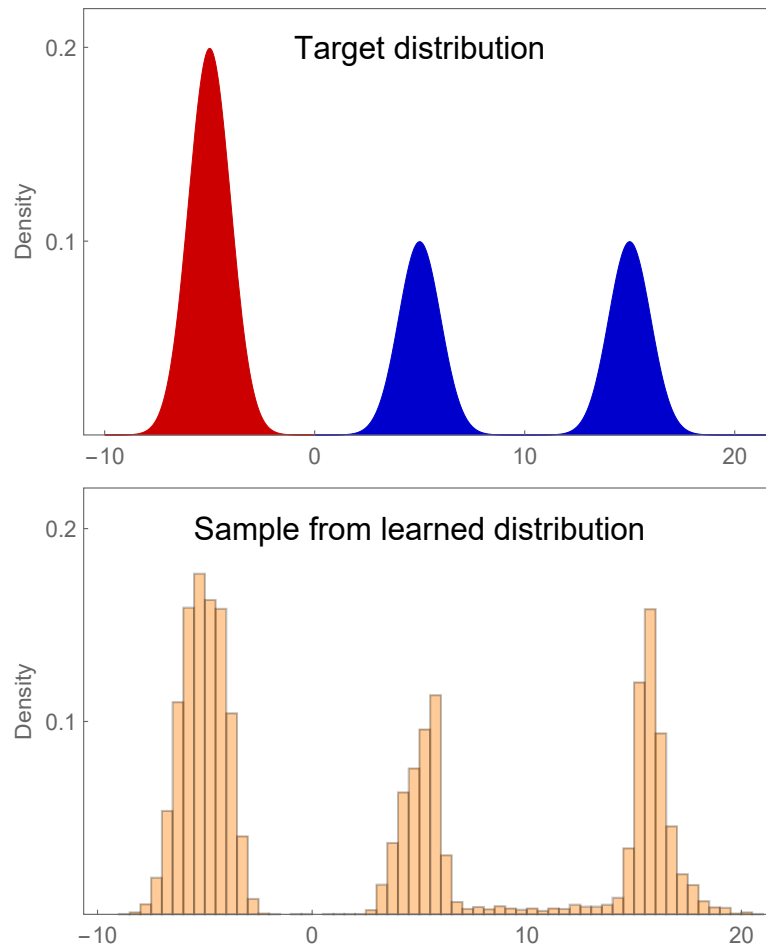
**Fig. 9.** Scatterplots of GAN-generated samples (red) for the bimodal bivariate target distribution, for training samples (blue) of three sizes  $N = 10$ , 100, and 1000. The GAN exhibits a strong learned bridge bias.

The univariate and bivariate bimodal target distributions in Figs. 6 and 9 have point modes and the GAN bridge between the modes is essentially one-dimensional. The Swiss roll distribution in our complexity experiment affords us an opportunity to see what can happen when a target mode extends beyond just a point. Figure 10 presents scatterplots of GAN-generated samples for the Swiss roll target distribution in our experiment. The training sample of size  $N = 10$  is not enough for the GAN to learn the submanifold on which the target ridge resides and it reverts to identifying modes with heavily biased bridges. With a relatively small training sample size  $N = 100$ , though, the GAN has discovered this structure and almost all bridge bias is gone. Bridge bias could be understood to arise from the fact that the generator is a continuous map from the latent space of the GAN input noise distribution to the support (data space) of the target distribution, while a bimodal target distribution with no bridge (zero mass between modes) would require a discontinuous map between the latent space and the data space. In other words, GAN bridge bias may be an artifact of a continuous approximation to a discontinuous function.



**Fig. 10.** Scatterplots of GAN-generated samples (red) for the Swiss roll target distribution, for training samples (blue) of three sizes  $N = 10$ , 100, and 1000. The GAN exhibits a bridge bias for  $N = 10$ . Unlike for our other one- and two-dimensional target distributions, though, the GAN soon learns the essential structure of the Swiss roll as  $N$  is increased, and bridge bias no longer occurs.

Our experiment found that GAN performance decreases as modes are added, and we identified bridge bias as a significant cause. In fact, a GAN bridge can exist between modes even in the absence of any corresponding target bridge. These GAN bridges may be the source of low-fidelity realizations synthesized by GANs trained for image generation [39]. A conditional GAN can alleviate bridge bias to some degree. A GAN is trained in an unsupervised fashion, with no labels or other identifying information attached to the examples in the training sample  $S$ . A conditional GAN is trained with labeled data, and if these class labels line up well with target modes, bridge bias can be reduced. To see this, consider the trimodal target distribution  $T$  shown (top) in Fig. 11. Training examples drawn from  $T$  are identified (red or blue) depending on the mode from which they derive. The color labels unambiguously identify the left mode, but they do not resolve the center and right target modes. This is akin to having labels for images of cats and dogs, but no labels specifying dog breed. A GAN trained with labeled data—a conditional GAN [40]—learned the distribution shown (bottom) in Fig. 11. The left and center modes from different classes have no appreciable bridge, while the center and right modes from the same class have a bridge. Of course, labeled data are not always available, and when labels are available they may not correspond to different modes. Still, further study is warranted to discover when and to what extent a conditional GAN with labeled training data can alleviate bridge bias.



**Fig. 11.** A trimodal target distribution (top) with color-labeled modes and data (bottom) synthesized by a conditional GAN. A bridge exists in the learned distribution between the center and right modes of the same labeled class but not between the left and center modes from different labeled classes.

#### 4. Summary and Related Remarks

GAN performance studies in the setting of high-dimensional applications have made only limited progress on pressing problems associated with GAN training, including mode collapse and training instability. In this study, to make progress on these and other questions relating to GAN performance, we took a fresh approach and considered GAN performance in low-dimensional settings. This approach offered important advantages: a reduced computational burden in experiments, more comprehensible and malleable target distributions, and easier assessment of GAN error. Our low-dimensional approach also carried risk. A low-dimensional GAN may not reproduce the high-dimensional phenomena that need to be understood. Encouragingly in this regard, our experiment revealed bridge bias in trained GANs, analogous to that seen in high dimensions.

A major purpose of our study was to establish protocols for GAN design and experimentation that fully exploit the advantages of low dimension and that can be used in subsequent, more elaborate experiments with low-dimensional GANs. Beyond this, our work makes two contributions. First, our work highlights the

perspective of GAN error as a function of training sample size, because our experiment shows that this relationship is log-log linear and that the GAN error exponent (log-log slope) depends solely on the dimension of the target distribution. Second, our experiment uncovers two prominent forms of GAN error, tail underfilling and bridge bias, finding that both decrease only slowly with increasing sample size.

This initial study of low-dimensional GAN performance offers a framework for further investigation in many directions. Target distribution complexity can be varied by dimension and by number, distribution, and dimensionality of modes. Our experiment found, for example, that doubling the dimension of the target support from 1 to 2 roughly halved (from 0.47 to 0.19) the GAN error exponent. Determining whether and how this effect scales to higher dimensions has important implications for defining how well GANs can reasonably be expected to perform in many application settings. Also, investigation into the relationship between bridge bias and bridge length (spacing between modes) is needed; all else remaining equal, modes tend to be more separated in higher-dimensional data spaces.

All the GAN training samples in our experiment were drawn from the target distribution by simple random sampling. Other sampling schemes can be envisioned; one such scheme is stratified random sampling, in which the training sample is assembled from simple random samples drawn from each target mode. This higher-fidelity sampling scheme could be expected to yield reduced GAN error for any given sample size  $N$ ; one would like to determine whether such a sampling scheme would, more powerfully, increase the GAN error exponent. The effect of training sample fidelity could be tested to its limit by studying training samples  $S$  that minimize the EM distance  $d(S, T)$  separating  $S$  and the target distribution  $T$ .

## Acknowledgments

The authors would like to thank Jake Rezac, Adam Pintar, and an anonymous reviewer, whose thoughtful comments and suggestions greatly improved the quality of this work.

## 5. References

- [1] Jin Y, Zhang J, Li M, Tian Y, Zhu H, Fang, Z (2017) Towards the automatic anime characters creation with generative adversarial networks. *arXiv preprint arXiv:1708.05509*. Available at <https://arxiv.org/abs/1708.05509>
- [2] Goodfellow I, Pouget-Abadi J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2018) Generative adversarial networks. *Advances in Neural Information Processing Systems* eds Jordan MI, LeCun Y, Solla SA (MIT Press, Cambridge, MA), pp 2672–2680.
- [3] Creswell A, White T, Dumoulin V, Arulkumaran K, Sengupta B, Bharath A (2018) Generative Adversarial Networks: An Overview. *IEEE Signal Processing Magazine* 35(1):53–65. <https://doi.org/10.1109/MSP.2017.2765202>
- [4] Putin E, Asadulaev A, Vanhaelen Q, Ivanenkov Y, Aladinskaya A, Aliper A, Zhavoronkov, A (2018) Adversarial Threshold Neural Computer for Molecular *de Novo* Design. *Molecular Pharmaceutics* 15(10):4386–4397. <https://doi.org/10.1021/acs.molpharmaceut.7b01137>
- [5] Mustafa M, Bard D, Bhimji W, Lukić Z, Al-Rfou R, Kratochvil JM (2019) CosmoGAN: Creating high-fidelity weak lensing convergence maps using Generative Adversarial Networks. *Computational Astrophysics and Cosmology* 6(1):1–13. <https://doi.org/10.1186/s40668-019-0029-9>
- [6] Hwang U, Choi S, Lee HB, Yoon S (2017) Adversarial training for disease prediction from electronic health records with missing data. *arXiv preprint arXiv:1711.04126*. Available at <https://arxiv.org/abs/1711.04126>
- [7] Radford, A, Metz L, Chintala S (2015) Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*. Available at <https://arxiv.org/abs/1511.06434>
- [8] Isola P, Zhu JY, Zhou T, Efros A (2017) Image-to-image translation with conditional adversarial networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp 1125–1134. Available at <https://arxiv.org/abs/1611.07004>
- [9] Karras T, Aila T, Laine S, Lehtinen J (2017) Progressive growing of GANs for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*. Available at <https://arxiv.org/abs/1710.10196>
- [10] Lim, Steffen and Khan, Sams and Alessandro, Matteo and McFall, Kevin (2019) Spatio-temporal super-resolution with photographic and depth data using GANs. *Proceedings of the 2019 ACM Southeast Conference*, pp 262–263. <https://doi.org/10.1145/3299815.3314482>



- [11] Zhu JY, Park T, Isola P, Efros A (2017) Unpaired image-to-image translation using cycle-consistent adversarial networks. *Proceedings of the IEEE International Conference on Computer Vision*, pp 2223–2232. <https://doi.org/10.1109/ICCV.2017.244>
- [12] Bau D, Zhu JY, Strobelt H, Zhou B, Tenenbaum JB, Freeman WT, Torralba A (2018) GAN dissection: Visualizing and understanding generative adversarial networks. *arXiv preprint arXiv:1811.10597*. Available at <https://arxiv.org/abs/1811.10597>
- [13] Lala S, Shady M, Belyaeva A, Liu M (2018) Evaluation of mode collapse in generative adversarial networks. *Proceedings of the IEEE High Performance Extreme Computing Conference*. Available at [http://www.ieee-hpec.org/2018/2018program/index\\_html\\_files/124.pdf](http://www.ieee-hpec.org/2018/2018program/index_html_files/124.pdf)
- [14] Theis L, van den Oord A, Bethge M (2015) A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*. Available at <https://arxiv.org/abs/1511.01844>
- [15] Salimans T, Goodfellow I, Zaremba W, Cheung V, Radford A, Chen X (2016) Improved Techniques for Training GANs. *Advances in Neural Information Processing Systems*, eds Jordan MI, LeCun Y, Solla SA (MIT Press, Cambridge, MA), pp 2234–2242.
- [16] Lee JD, Simchowitz M, Jordan MI, Recht B (2016) Gradient descent only converges to minimizers. *Proceedings of Machine Learning Research* 49:1246–1257. Available at <http://proceedings.mlr.press/v49/lee16.pdf>
- [17] Arjovsky M, Bottou L (2017) Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*. Available at <https://arxiv.org/abs/1701.04862>
- [18] Arora S, Ge R, Liang Y, Ma T, Zhang Y (2017) Generalization and equilibrium in generative adversarial nets (GANs). *Proceedings of Machine Learning Research* 70:224–232. Available at <http://proceedings.mlr.press/v70/arora17a/arora17a.pdf>
- [19] LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11):2278–2324. <https://doi.org/10.1109/5.726791>
- [20] Arjovsky M, Chintala S, Bottou L (2017) Wasserstein generative adversarial networks. *Proceedings of Machine Learning Research* 70:214–223. Available at <http://proceedings.mlr.press/v70/arjovsky17a/arjovsky17a.pdf>
- [21] Nagarajan V, Kolter JZ (2017) Gradient Descent GAN Optimization is locally stable. *Advances in Neural Information Processing Systems*, eds Jordan MI, LeCun Y, Solla SA (MIT Press, Cambridge, MA), pp 5585–5595.
- [22] Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, Courville AC (2017) Improved Training of Wasserstein GANs. *Advances in Neural Information Processing Systems*, eds Jordan MI, LeCun Y, Solla SA (MIT Press, Cambridge, MA), pp 5767–5777.
- [23] Sønderby CK, Caballero J, Theis L, Shi W, Huszár F (2016) Amortised map inference for image super-resolution. *arXiv preprint arXiv:1610.04490*. Available at <https://arxiv.org/abs/1610.04490>
- [24] Mescheder L, Geiger A, Nowozin S (2018) Which training methods for GANs do actually converge?. *International Conference on Machine Learning*, pp 3481–3490. Available at <https://arxiv.org/abs/1801.04406>
- [25] Dobrushin RL (1970) Prescribing a system of random variables by conditional distributions. *Theory of Probability & Its Applications* 15(3):458–486. <https://doi.org/10.1137/1115049>
- [26] Mescheder L, Nowozin S, Geiger A (2017) The Numerics of GANs. *Advances in Neural Information Processing Systems*, eds Jordan MI, LeCun Y, Solla SA (MIT Press, Cambridge, MA), pp 1825–1835.
- [27] Kingma DP, Ba J (2014) Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. Available at <https://arxiv.org/abs/1412.6980>
- [28] Monge G (1781) Mémoire sur la théorie des déblais et des remblais. *Histoire de l'Académie Royale des Sciences de Paris* (De l'Imprimerie Royale, Paris, France).
- [29] Peleg S, Werman M, Rom H (1989) A unified approach to the change of resolution: Space and gray-level. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11(7):739–742. <https://doi.org/10.1109/34.192468>
- [30] Rubner Y, Tomasi C, Guibas LJ (1998) A metric for distributions with applications to image databases. *Sixth International Conference on Computer Vision*, pp 59–66. <https://doi.org/10.1109/ICCV.1998.710701>
- [31] Grauman K, Darrell T (2004) Fast contour matching using approximate earth mover's distance. *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. <https://doi.org/10.1109/CVPR.2004.1315035>
- [32] Cha SH (2007) Comprehensive survey on distance/similarity measures between probability density functions. *International Journal of Mathematical Models and Methods in Applied Sciences* 1(4):300–307. Available at <https://www.naun.org/main/NAUN/ijmmas/mmmas-49.pdf>
- [33] Borji A (2019) Pros and cons of GAN evaluation measures. *Computer Vision and Image Understanding* 179:41–65. <https://doi.org/10.1016/j.cviu.2018.10.009>
- [34] Cuturi M (2013) Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in Neural Information Processing Systems*, eds Jordan MI, LeCun Y, Solla SA (MIT Press, Cambridge, MA), pp 2292–2300. Available at <https://arxiv.org/abs/1306.0895>
- [35] Auricchio G, Bassetti F, Gualandi S, Veneroni, M (2018) Computing Kantorovich-Wasserstein distances on  $d$ -dimensional histograms using  $(d + 1)$ -partite graphs. *Advances in Neural Information Processing Systems*, eds Jordan MI, LeCun Y, Solla SA (MIT Press, Cambridge, MA), pp 5793–5803. Available at <https://arxiv.org/abs/1805.07416>
- [36] Flamary R, Courty N (2019) Pot python optimal transport library. Available at <https://pythonot.github.io/>
- [37] Marsland S (2015) Machine Learning: An Algorithmic Perspective (CRC Press, Boca Raton, FL).
- [38] Kutner MH, Nachtsheim CJ, Neter J, Li W (2005) Applied Linear Statistical Models, Vol. 5 (McGraw-Hill Irwin, Boston, MA).
- [39] Brock A, Donahue J, Simonyan K (2018) Large scale GAN training for high fidelity natural image synthesis. *International*

- Conference on Learning Representations*. Available at <https://arxiv.org/abs/1809.11096>
- [40] Mirza M, Osindero S (2014) Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*. Available at <https://arxiv.org/abs/1411.1784>

**About the authors:** *Felix Jimenez, Amanda Koepke, Mary Gregg, and Michael Frey are all members of the Statistical Engineering Division at the National Institute of Standards and Technology.*

*Prior to joining NIST, Felix Jimenez was a PREP researcher with the University of Colorado-Boulder. He is currently pursuing his Ph.D. degree at Texas A&M University.*

*Amanda Koepke received her Ph.D. degree in statistics from the University of Washington in 2014.*

*Mary Gregg is a National Research Council postdoctoral research associate.*

*Michael Frey is the group leader of NIST's Statistical Engineering Division in Boulder.*

*The National Institute of Standards and Technology is an agency of the U.S. Department of Commerce.*