SIAM J. MATH. DATA SCI.
Vol. 3, No. 4, pp. 1197–1222

MR-GAN: Manifold Regularized Generative Adversarial Networks for Scientific Data*

Qunwei Li[†], Bhavya Kailkhura[‡], Rushil Anirudh[‡], Jize Zhang[‡], Yi Zhou[§], Yingbin Liang[¶], T. Yong-Jin Han[‡], and Pramod K. Varshney[‖]

Abstract. Despite the growing interest in applying generative adversarial networks (GANs) in complex scientific applications, training GANs on scientific data remains a challenging problem from both theoretical and practical standpoints. One reason for this is that the generator is unable to accurately capture the underlying complex manifold structure of the real scientific data using only gradients from the discriminator. In this paper, we address this challenge using a novel approach that exploits the unique geometry of the scientific data to improve the quality of the generated data. Specifically, we improve the training of the GAN using an additional term referred to as a manifold regularizer which encourages the generator to respect the unique geometry of the scientific data manifold and generate high quality data. We theoretically prove that the addition of this regularization term leads to improved performance for different classes of GANs including deep convolutional GAN and Wasserstein GAN. Finally, we carry out performance comparisons on diverse datasets: synthetic data (Gaussian mixture), natural image data (celebrity face images (CelebA)), and scientific experimental data (scanning electron microscopy images of organic crystalline materials). In most of these applications, we find that the proposed manifold regularization-based approach helps in avoiding mode collapse, produces stable training, and leads to significant gains in terms of *geometry score* compared to its unregularized counterparts.

Key words. generative modeling, manifold, GANs, scientific ML, unsupervised

[*] Received by the editors June 9, 2020; accepted for publication (in revised form) July 6, 2021; published electronically November 8, 2021.

https://doi.org/10.1137/20M1344299

Funding: The work of the sixth author was supported in part by the U.S. National Science Foundation under the grant CCF-1900145. This work was performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344, Lawrence Livermore National Security, LLC. This document was prepared as an account of the work sponsored by an agency of the United States Government. Neither the United States Government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or Lawrence Livermore National Security, LLC. The views and opinions of the authors expressed herein do not necessarily state or reflect those of the United States Government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes. This work was supported by LLNL Laboratory Directed Research and Development project 19-SI-001 and released with LLNL tracking number LLNL-CONF-798660.

[†] Ant Financial Services Group, China (qunwei.qw@antfin.com).

[‡] Lawrence Livermore National Laboratory, Livermore, CA 94550 USA (kailkhura1@llnl.gov, anirudh1@llnl.gov, zhang64@llnl.gov, han5@llnl.gov).

[§] University of Utah, Salt Lake City, UT 84112 USA (yi.zhou@utah.edu).

[¶] The Ohio State University, Columbus, OH 43210 USA (liang.889@osu.edu).

[‖] Syracuse University, Syracuse, NY 13244 USA (varshney@syr.edu).

AMS subject classifications. 68T99, 90C47, 91A40

DOI. 10.1137/20M1344299

1. Introduction.

Machine learning (ML) provides incredible opportunities in a wide range of scientific applications, such as material science [14], cosmology [33], and medicine [7]. One application of ML that has emerged in recent years is the use of generative adversarial networks (GANs) [10] to produce synthetic data that emulates real scientific data [42, 6, 41, 43, 30]. The core of the training of GANs is a min-max game in which two neural networks (generator and discriminator) compete with each other: the generator tries to trick the discriminator/classifier into classifying its generated synthetic/fake data as true. These synthetic samples then can be used to overcome the limitations caused by small datasets and limited amount of annotated samples [8], data quality enhancement [20], data analysis and model introspection [23], etc.

Despite the interest that GANs have drawn, the task of training GANs (on both scientific and natural data) remains a challenging problem, both from theoretical and practical standpoints. Specifically, GAN training suffers from the following major problems: (a) *mode-collapse*: the generator collapses which results in a poor generalization, i.e., producing limited varieties of samples; (b) *lack of equilibrium*: the min-max game may not have any equilibrium; and (c) *instability*: even when the equilibrium exists, model parameters may oscillate, destabilize, and never converge to an equilibrium. These failure modes result in generation of poor quality data especially in scientific applications.

It was shown in [1] that the real data lies in a submanifold of the Euclidean space, and the generated data and the real data lying in disjoint manifolds is one of the reasons for the aforementioned problems in the training of GANs. Scientific data, in particular, need not lie on simple smooth manifolds like most natural images. As a result, most commonly made assumptions about images are likely to fail when it comes to scientific datasets. Consequently, it is imperative to incorporate intrinsic manifold information in order to accurately capture the complex geometries of scientific datasets. Motivated by this insight, this paper takes some initial steps towards designing GAN architectures which can exploit the unique geometry of the real data, such as its manifold structure, to overcome the aforementioned problems. The basic idea is simple yet powerful: in addition to the gradient information provided by the discriminator, we want the generator to exploit other geometric information present in the real data, such as the manifold information. Taking advantage of this additional information, we will have more stable gradients while training our generator. Specifically, we propose a novel method for incorporating geometry and regularizing the GAN training by adding an additional regularization term—called the manifold regularizer—with generator updates. The proposed manifold regularizer forces the generator to respect the unique geometry of the real data manifold. We prove theoretically that the addition of this regularization term in any class of GANs (including deep convolutional GAN (DCGAN) and Wasserstein GAN) leads to improved performance. In practice, the manifold regularized GANs (MR-GANs) are simple to implement and result in improved quality of generated data in a wide range of computer vision and scientific applications compared to their unregularized counterparts.

1.1. Related work.

In the literature, the unstable behavior of GAN training is relatively understudied, with few notable exceptions like the remarkable work in [1]. The authors provide important insights into mode collapse and instability in GAN training. They show that these issues arise when the supports of the generated distribution and the true distribution are disjoint. The authors in [3], on the other hand, explore questions relating to the sample complexity and expressiveness of the GAN architecture and their relation to the existence of an equilibrium. Given that an equilibrium exists, the convergence of GANs with an update procedure using gradient descent was studied in [31]. The estimation and generalization errors of GAN training was considered in [13]. The authors in [12] investigated the minimax estimation problem of the neural net distance and justified the empirical neural net distance as a good approximation of the true neural net distance for training GANs in practice.

From a practical perspective, various architectures and training objectives have been proposed to address GAN training challenges [2, 36, 16]. The authors in [22] improved the generative moment matching network and used maximum mean discrepancy with adversarially learned kernels to have better hypothesis testing power. Several optimization heuristics and architectures have also been proposed to address challenges such as mode collapse [40, 27, 37, 5]. Methods for regularizing the discriminator for better stability were devised in [38, 26, 31, 18, 28]. The authors in [38] presented a stabilizing regularizer that is based on a gradient norm, where the gradient is calculated with respect to the data samples. The authors in [28] proposed a weight normalization technique called spectral normalization to stabilize the training of the discriminator. Other regularization approaches to improve GAN training can be found in [21]. On the other hand, the authors of [26, 31] designed regularizers based on the norm of a gradient calculated with respect to the parameters. The authors in [18] applied a Jacobian regularizer to the discriminator of a feature-matching GAN to improve the performance of GAN-based semisupervised learning. In contrast to regularizing the discriminator, this paper proposes to regularize the generator for improving GAN training. Finally, the authors in [35] proposed replacing the original GAN loss with a different loss function matching the statistical mean and radius of the spheres approximating the geometry of the real data and generated data. However, characterizing the geometric information of the data only by the mean and the radius of the sphere representing the data loses a significant amount of geometrical information. The construction in [35] was purely heuristic and did not have any theoretical backing. On the contrary, we directly exploit the undistorted manifold information for regularizing the training of the generator rather than treating it as a loss function and theoretically prove that the proposed approach yields improved performance.

On the application front, there is a surge of interest in exploiting GANs in a wide range of scientific applications. The authors in [42] and [24] applied GANs for the microstructural materials design and analysis problem. Application of GANs to the high energy particle physics problem was explored by [6]. In [41], the authors trained a GAN model to generate images resembling the iconic Hubble Space Telescope Extreme Deep Field offering a new data-driven approach for producing realistic mock surveys and synthetic data at scale, in astrophysics. A survey of application of GAN in healthcare was provided in [43].

1.2. Contributions.

The main contributions of the paper are summarized as follows:

- We propose a novel method for regularizing GAN training by incorporating an additional regularization term that respects the unique geometry of the real data manifold.

- We prove that the proposed training objective function can be realized with a sufficiently small bias using deep neural networks (DNNs).
- We show that the equilibrium of the min-max game for the proposed MR-GANs exists and can be attained by DNNs in practice.
- We prove that the training of the proposed MR-GAN is exponentially stable around the equilibrium.
- We show empirically in a wide range of computer vision and scientific applications that MR-GANs are able to avoid model collapse and significantly outperform several widely used baseline GAN architectures.

Finally, manifold regularization in MR-GAN is extremely simple in that it can be implemented using a few lines of Python code and added to any existing GAN implementation to improve its performance (see Appendix F).

2. Preliminaries.

In this section, we give a brief introduction of GANs and manifold learning. We will also briefly discuss how manifold learning principles can be exploited to have a better GAN formulation.

2.1. Introduction of GANs.

Throughout the paper, we use d for the dimension of samples, p for the number of parameters in generator/discriminator, and m for the number of samples. Let $\{G_u, u \in \mathcal{U}\}$, with $\mathcal{U} \in \mathbb{R}^p$, denote the class of generators, where G_u is a function—which is often a neural network in practice—from $\mathbb{R}^l \to \mathbb{R}^d$ indexed by u that denotes the parameters of the generators. Here $\mathcal{U}$ denotes the possible ranges of the parameters, and without loss of generality we assume that $\mathcal{U}$ is a subset of the unit ball. The generator G_u defines a distribution $\mathcal{D}_{G_u}$ as follows: generate h from an l-dimensional spherical Gaussian distribution, apply G_u on h, and generate a sample $x = G_u(h)$ from the distribution $\mathcal{D}_{G_u}$. We drop the subscript u in $\mathcal{D}_{G_u}$ when it is clear from the context. Let $\{D_v, v \in \mathcal{V}\}$ denote the class of discriminators, where D_v is a function from $\mathbb{R}^d$ to $[0, 1]$ and v is the parameter of D_v. Training the discriminator consists of making its output a high value (preferably 1) when x is sampled from the distribution $\mathcal{D}_{real}$ and a low value (preferably 0) when x is sampled from the synthetic distribution $\mathcal{D}_{G_u}$. On the contrary, training the generator consists of making its synthetic distribution "similar" to $\mathcal{D}_{real}$ in the sense that the discriminator's output tends to indicate that the two distributions are close.

The original GAN training problem [10] is formulated as the following min-max game between the generator and the discriminator:

$$\min_{u\in\mathcal{U}} \max_{v\in\mathcal{V}} \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} [\log D_v(x)] + \mathop{\mathbb{E}}_{y\sim\mathcal{D}_{G_u}} [\log(1 - D_v(y))].$$

Intuitively, this forces the discriminator D_v to give high values $D_v(x)$ to the real samples and low values $D_v(y)$ to the generated examples. The log function is used because of its interpretation as the likelihood. However, in practice this formulation may not provide sufficient gradient for the generator to learn well, as the term $[\log(1-D_v(y))]$ may saturate early during the training process. Therefore, we consider a more general formulation by using a monotone function $\phi : [0, 1] \to \mathbb{R}$, which yields the following objective[1]:

[1] Note that this form of the objective function has connections with integral probability metrics-based training of GANs [29].

$$\min_{u\in\mathcal{U}}\max_{v\in\mathcal{V}} \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \mathop{\mathbb{E}}_{y\sim\mathcal{D}_{G_u}} [\phi(D_v(x)) + \phi(1-D_v(y))].$$

We call the function ϕ the measuring function. It should be concave so that when $\mathcal{D}_{real}$ and $\mathcal{D}_{G_u}$ are the same distributions, the best strategy for the discriminator is simply to output $1/2$ and the optimal value is $2\phi(1/2)$. In later proofs, we require ϕ to be bounded and Lipschitz. In practice, training often uses $\phi(x) = \log(\delta+(1-\delta)x)$ (which takes values in $[\log\delta, 0]$ and is $1/\delta$-Lipschitz), and the recently proposed Wasserstein GAN [2] objective function uses $\phi(x) = x$ (which takes values in $[0, 1]$ (by definition) and is 1-Lipschitz).

2.2. Manifold learning. In several ML applications, the data lies on or close to the surface of one or more low-dimensional manifolds embedded in the high-dimensional ambient space. Attempting to uncover this manifold structure in a dataset is referred to as manifold learning [4].

Given a set $\mathbf{X}$ of data (or feature) vectors, a graph $\mathcal{G} = \{\mathbf{X}, \mathbf{\Omega}\}$ is used to characterize the manifold-based relationships among these vectors. Here, $\mathbf{\Omega} = [w_{ij}]$ is a matrix containing the weights over edges connecting graph nodes and is referred to as the affinity matrix. The weight, w_{ij}, on an edge connecting two nodes, $\mathbf{x}_i$ and $\mathbf{x}_j$, provides a measure of closeness between them. These weights govern various characteristics of a graph, including structure, connectivity, and compactness. Graph-based relationships are usually characterized using the Euclidean distance based Gaussian heat kernel given by

$$w_{ij} = \exp\left(\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{\rho}\right), \tag{2.1}$$

where ρ is the kernel scale parameter. One can also use

$$w_{ij} = \begin{cases} \exp\left(\frac{-\|\mathbf{x}_i-\mathbf{x}_j\|^2}{\rho}\right), & e(\mathbf{x}_i, \mathbf{x}_j) = 1, \\ 0 & \text{otherwise,} \end{cases} \tag{2.2}$$

where the function $e(\mathbf{x}_i, \mathbf{x}_j)$ indicates whether $\mathbf{x}_i$ lies near the predefined neighborhood of $\mathbf{x}_j$. As an example, given input $\mathcal{G} = \{\mathbf{X}, \mathbf{\Omega}\}$, manifold learning inspired learning approaches attempt to constrain the output, $\mathbf{z} = f(\mathbf{X})$,[2] to preserve the structure (compactness) in $\mathbf{X}$ (defined by the affinity weight matrix $\mathbf{\Omega}$). This is usually achieved by employing a regularization term along with a task-specific loss function. A case of particular recent interest in manifold regularized learning is when the support of the data is a compact submanifold $\mathcal{M} \in R^d$. In that case, one natural choice for the regularizer is $\int_{x\in\mathcal{M}} \|\nabla_{\mathcal{M}}\mathbf{z}\|^2 dP_X(x)$, where $\nabla_{\mathcal{M}}$ is the gradient of $\mathbf{z}$ along the manifold $\mathcal{M}$ and the integral is taken over the marginal distribution. In most applications, the marginal P_X is not known. Therefore, we need to get empirical estimates of P_X and the regularizer. The term $\int_{x\in\mathcal{M}} \|\nabla_{\mathcal{M}}\mathbf{z}\|^2 dP_X(x)$ may be approximated on the basis of data samples using the graph Laplacian L associated to the data, which yields an estimate $\frac{1}{m^2} Tr(\mathbf{z}^T L\mathbf{z})$, where $\mathbf{z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_m]$. This estimate simplifies to $\sum_{i,j} \|\mathbf{z}_i - \mathbf{z}_j\|^2 w_{ij}$ as $L = \mathbf{D} - \mathbf{\Omega}$, where $\mathbf{D}$ is the diagonal matrix with $\mathbf{D}_{ii} = \sum_j w_{ij}$.

[2] Here the model $f(\cdot)$ and corresponding output $\mathbf{z}$ depend on the task of interest, e.g., supervised learning or generative modeling.

3. MR-GAN.

3.1. Geometry-aware GANs.

A reasonable approach for GANs would be to use the conventional manifold regularizer $Tr(\mathbf{y}^T L\mathbf{y})$ at the generator to force the generated data $\mathbf{y}$ to respect the geometry of the real data $\mathbf{X}$. However, our initial experiments suggested that the conventional regularizer does not perform well in practice. In fact, with the conventional manifold regularizer at the generator, our theoretical analysis also indicates that the equilibrium cannot be guaranteed. Therefore, we propose a novel regularizer at the generator to force the generated data to respect the geometry of the real data. Furthermore, for this new formulation, we theoretically show that some of the issues with GAN training can be overcome.

3.2. Proposed GAN architecture.

Motivated by the considerations above, in this section we propose a novel regularization penalty for the generator updates, which employs a term based on the gradient of the embedding function ψ in the intrinsic manifold, to incorporate the fact that the real data is indeed extremely concentrated on a low-dimensional manifold [32]. The embedding function ψ serves two purposes. First, it extracts useful information from the raw data for better inference. Second, it is a dimension-reduction mapping, which can prevent overfitting during training. As we will show later, the regularization term does not change the parameter values at the equilibrium point, and it further enhances the local stability of the optimization procedure. Specifically, we propose the following regularized objective of MR-GAN[3] as follows:

$$\begin{aligned}\min_{u\in\mathcal{U}}\max_{v\in\mathcal{V}} \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \mathop{\mathbb{E}}_{y\sim\mathcal{D}_{G_u}} &[\phi(D_v(x)) + \phi(1-D_v(y)) \\ &+ \lambda \int_{x\sim\mathcal{M}} \|\nabla_{\mathcal{M}}(\psi(y)-\psi(x))\|^2 dPx],\end{aligned} \tag{3.1}$$

where ψ is an embedding function which takes the form $\psi : x \to \tilde{x}$, and $\tilde{x}$ lies within a manifold embedded in $\mathbb{R}^d$. In our experiments, we use either an autoencoder or the identity mapping as our embedding function ψ. Essentially, the regularizer is the squared magnitude of the gradient of the embedding function in the intrinsic manifold, with respect to the difference between the real and generated data. When the support of distribution $\mathcal{D}_{real}$ lies in the manifold $\mathcal{M}$, the objective (3.1) becomes the following because we have an expectation operator over the distribution of the real data:

$$\begin{aligned}\min_{u\in\mathcal{U}}\max_{v\in\mathcal{V}} \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \mathop{\mathbb{E}}_{y\sim\mathcal{D}_{G_u}} &[\phi(D_v(x)) + \phi(1-D_v(y)) \\ &+ \lambda\|\nabla_{\mathcal{M}}(\psi(y)-\psi(x))\|^2].\end{aligned} \tag{3.2}$$

We show later that the proposed MR-GAN architecture enjoys provable performance guarantees.

3.3. Manifold regularized training.

We provide intuitions that the objective function of the proposed MR-GAN helps in aligning the manifold of the generated data with the manifold of the real data. Let us denote the objective function of MR-GAN as

[3] Please see (3.7) for an empirical version of the MR-GAN formulation.

$$\begin{aligned} F(u,v) = &\mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \mathop{\mathbb{E}}_{y\sim\mathcal{D}_{G_u}} [\phi(D_v(x)) + \phi(1 - D_v(y)) \\ &+ \lambda\|\nabla_{\mathcal{M}}(\psi(y) - \psi(x))\|^2]. \end{aligned} \tag{3.3}$$

Regularized gradients. Note that the gradient for the generator of MR-GAN is given by

$$\begin{aligned} \frac{\partial F(u,v)}{\partial u} = &\mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \int_{\mathcal{Y}} \nabla_u(p_u(y)\phi(1 - D_v(y)) \\ &+ \lambda\|\nabla_{\mathcal{M}}(\psi(y) - \psi(x))\|^2)dy, \end{aligned} \tag{3.4}$$

where $\mathcal{Y}$ is the domain of the generated samples and p_u is the probability density function of the distribution $\mathcal{D}_{G_u}$ for the generated samples and is dependent on u. The first term $\mathbb{E}_{x\sim\mathcal{D}_{real}} \int_{\mathcal{Y}} \nabla_u p_u(y)\phi(1 - D_v(y,\tilde{x}))dy$ follows the geometric properties of the measuring function ϕ. When the manifold $\mathcal{M}_{\mathcal{Y}}$ (where the support of $\mathcal{Y}$ lies) and the manifold $\mathcal{M}$ are far away, $\|\nabla_{\mathcal{M}}(\psi(y) - \psi(x))\|^2$ is very large. This should strongly drive $\mathcal{M}_{\mathcal{Y}}$ to $\mathcal{M}$. When $\mathcal{M}_{\mathcal{Y}}$ and $\mathcal{M}$ become closer, $\|\nabla_{\mathcal{M}}(\psi(y) - \psi(x))\|^2$ is smaller. This resembles L^2 optimization in general, where the loss function offers an adaptive gradient toward the optima. The gradient $\nabla_{\mathcal{M}}$ provides a multimodal weighting, and the modes of $\mathcal{D}_{real}$ will thus drive the gradient in training the generator.

Bounded objective function. Additionally, recalling the objective function (3.1), the regularizer plays a role in the training of the generator, which has the form

$$\min_{u\in\mathcal{U}} \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \mathop{\mathbb{E}}_{y\sim\mathcal{D}_{G_u}} [\phi(1 - D_v(y)) + \lambda\|\nabla_{\mathcal{M}}(\psi(y) - \psi(x))\|^2]. \tag{3.5}$$

If the embedding function ψ is L_ψ-Lipschitz smooth on the manifold (which we assume in what follows), we have the following inequality:

$$\|\nabla_{\mathcal{M}}(\psi(y) - \psi(x))\|^2 \leq L_\psi^2\|y - x\|^2, \tag{3.6}$$

and further we can obtain

$$\begin{aligned} &\min_{u\in\mathcal{U}} \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \mathop{\mathbb{E}}_{y\sim\mathcal{D}_{G_u}} [\phi(1-D_v(y))+\lambda\|\nabla_{\mathcal{M}}(\psi(y)-\psi(x))\|^2 \\ &\quad\leq \min_{u\in\mathcal{U}} \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \mathop{\mathbb{E}}_{y\sim\mathcal{D}_{G_u}} \phi(1 - D_v(y)) + \lambda L_\psi^2\|y - x\|^2]. \end{aligned}$$

The regularization term $\lambda L_\psi^2\|y - x\|^2$ imposes the similarity between the generated and the real data, and thus our method penalizes dissimilarity between the generated and the real data. Essentially, our method finds the generated data closer to the real one and incorporates the geometric information of the real data into the data being generated. Later we show that y does not overfit to x, and y generalizes well with the proposed GAN architecture.

Training practices. The objective function (3.1) (or (3.3)) assumes that we have an infinite number of samples from $\mathcal{D}_{real}$ to estimate the value $\mathbb{E}_{x\sim\mathcal{D}_{real}}[\phi(D_v(x,\tilde{x}))]$. In practice, the objective function $F(u,v)$ is approximated with a finite number of training samples, which is denoted by $\hat{F}(u,v)$ and is expressed as

$$\hat{F}(u,v) = \frac{1}{m}\sum_{i=1}^{m} \phi(D_v(x_i)) + \phi(1 - D_v(y_i))$$
$$+ \frac{\lambda}{m^2}\sum_{i=1,j=1}^{m} \|\psi(y_i) - \psi(x_i) - \psi(y_j) + \psi(x_j)\|^2 w_{ij}. \tag{3.7}$$

With finite training examples $x_1, \dots, x_m \sim \mathcal{D}_{real}$, one uses $\frac{\lambda}{m}\sum_{i=1}^{m} \phi(D_v(x_i))$ in practice to estimate the quantity $\mathbb{E}_{x\sim\mathcal{D}_{real}}[\phi(D_v(x))]$. Similarly, one can use an empirical version to estimate $\mathbb{E}_{x\sim\mathcal{D}_{real}}\mathbb{E}_{y\sim\mathcal{D}_{G_u}}\phi(1 - D_v(y))$. Based on well-known manifold learning results, the regularization term $\|\nabla_{\mathcal{M}}(\psi(y) - \psi(x))\|^2$ can be approximated as

$$\frac{1}{m^2}\sum_{i=1,j=1}^{m} \|\psi(y_i) - \psi(x_i) - \psi(y_j) + \psi(x_j)\|^2 w_{ij}. \tag{3.8}$$

We use (2.1) in our implementation to estimate w_{ij} applied to the encoded data obtained using the embedding function, e.g., autoencoder or identity function.

Note that the first term on the right-hand side of (3.7) is the conventional objective function to train a GAN. The second term is associated with two different paired-up batch samples and training outcomes.

Here, using the definition of w_{ij}, we observe that if the data samples x_i and x_j are from different submanifolds, it encourages the output y_i and y_j to lie in different manifolds. Additionally, if x_i and x_j are from the same submanifold, it encourages the output y_i and y_j to lie in the same manifold. The regularizer helps in exploiting the information regarding inter- and intrarelations of the modes of the distribution of the real data and couples x_i and y_i in a manifold learning fashion.

4. Theoretical analysis.

This section provides provability results and properties of MR-GAN.[4] We first discuss the assumptions that we make in our analysis, which are widely used in the analysis of GANs (or DNNs) [39, 3, 31].

4.1. Assumptions.

Assumption 1. We make the following assumptions of G_u, D_v, and ψ.

(a) $\forall u, u' \in \mathcal{U}$ and any input h, $\|G_u(h) - G_{u'}(h)\| \le L\|u - u'\|$.

(b) $\forall u \in \mathcal{U}$ and any input h and h', $\|G_u(h) - G_u(h')\| \le L'\|h - h'\|$.

(c) The embedding function ψ is L_ψ-Lipschitz smooth on manifold $\mathcal{M}$, i.e., $\|\nabla_{\mathcal{M}}\psi(y) - \nabla_{\mathcal{M}}\psi(x)\|^2 \le L_\psi^2\|y - x\|^2$.

(d) $\mathcal{D}_{G_{u^*}} \sim \mathcal{D}_{real}$ and $D_{v^*} = 0\ \forall x \in supp(\mathcal{D}_{real})$.

(e) $\exists_{\epsilon_G} > 0$ such that $\forall u \in B_{\epsilon_G}(u^*)\ supp(\mathcal{D}_{G_u}) = supp(\mathcal{D}_{real})$, where $B_\epsilon(\cdot)$ denotes the l_2-ball of radius ϵ.

Assumption 1(a) means that G_u is L-Lipschitz with respect to its parameters, and we assume so for D_v as well. Note that this is distinct from the assumption that functions G_u, D_v are Lipschitz (which we introduce next) which focuses on the change in function value when we change x while keeping u, v fixed. Assumption 1(b) means that G_u is L'-Lipschitz

[4] The proofs are provided in the appendix.

with respect to its input, and we assume so for D_v as well. It essentially means that a small variation in the input to the generator/discriminator does not cause a large variation in the output of the generator/discriminator. Assumption 1(c) assumes the smoothness of the embedding function on manifold $\mathcal{M}$. The embedding function in practice could be an auto-encoder, which also satisfies this condition. Assumption 1(d) and Assumption 1(e) are "realizability" and "same support" conditions from [31].

Assumption 2. We make the following assumptions about the measure function ϕ.

(a) $\forall x \in \mathcal{M}, \|\nabla_{\mathcal{M}}\phi(x)\| \leq M$.
(b) The function ϕ is bounded in $[-\Delta, \Delta]$ in training.
(c) $\forall x, x' \in \mathbb{R}$, $\|\phi(x) - \phi(x')\| \leq L_\phi \|x - x'\|$.
(d) $\nabla_v^2 F(u^*, v)$ evaluated at v^* is negative definite, and $\nabla_u^2 \|\nabla_v F(u, v^*)\|^2$ evaluated at u^* is positive definite, where $F(\cdot)$ is defined in (3.3).

Assumption 2(a) is equivalent to the fact that the function ϕ has no geometrically step-sized property in function values. The training of the original GAN and Wasserstein GAN uses $\phi(x) = \log(\delta + (1-\delta)x)$ and $\phi(x) = x$, respectively. Also, $\log(x)$ and x are not bounded by nature when $x \in \mathbb{R}$. However, since ϕ takes input from $[0,1]$, Assumption 2(b) is valid. Assumption 2(c) implies that the measure function ϕ is L_ϕ-Lipschitz continuous. Assumption 2(d) is the "strong curvature" condition from [31].

4.2. Analytical results.

Generalization. Since we can only access (and optimize) the empirical distance between the distributions in practice, it becomes important to ensure that this empirical distance is close to the true distance for the generated and the real distributions. As the training algorithm is supposed to run in polynomial time, one has to estimate the true distance using only a polynomial number of samples [3]. Indeed, it is shown in [3] that if we do not have enough samples for training the GAN, (1) the distance between the empirical distributions can be close to the maximum possible distance even if the samples are drawn from the same distribution and (2) even if the generator happens to find the real distribution, the distance between the empirical distributions can still be large and the generator has no idea that it has succeeded.

Thus, it is crucial to answer the following question: can MR-GAN approximate the true distance between the generated and the real distributions with a reasonable number of samples?

Theorem 4.1. *Let $\hat{\mathcal{D}}_{real}$ and $\hat{\mathcal{D}}_{G_u}$ be empirical versions with at least m samples each for the MR-GAN. Then, there exists a universal constant C such that when*

$$m \geq \frac{Cp\log(LL_\phi p/\epsilon)(\Delta + 4\lambda M^2)^2}{\epsilon^2},$$

we have, with probability at least $1 - \exp(1-p)$,

$$|F(u,v) - \hat{F}(u,v)| \leq \epsilon. \tag{4.1}$$

Proof. See Appendix B. ■

The above theorem shows that if a sufficient amount of training data is available, the distance between the empirical objective function $\hat{F}(u,v)$ and the population objective function

$F(u,v)$ is sufficiently small. Although this does not directly imply the ability of the GAN to model the true data generating distribution, this result is important as it guarantees that the analysis of MR-GAN conducted based on the population objective function can be well generalized to the empirical form. Thus, it ensures that the theoretical guarantees of MR-GAN can be well satisfied in practice.

Existence of equilibrium. The training of GANs has the goal to end up with an equilibrium for the min-max game between the generator and the discriminator. That is, the discriminator outputs 1/2 for both the cases where the input is the real data or the generated data, which essentially means that the discriminator guesses randomly and cannot distinguish between real and generated data. On the other hand, the generator cannot exploit the output of the discriminator by back-propagation and cannot update itself and improve the quality of the generated data anymore. Therefore, ensuring the existence of the equilibrium of a certain GAN architecture is crucial before training process starts. It is important that we have provable results showing the existence of equilibrium for the proposed MR-GAN. Interestingly, if we change the regularizer in the objective function (3.2) from $\|\nabla_{\mathcal{M}}(\psi(y_i)-\psi(x_i))\|^2$ to $\|\nabla_{\mathcal{M}}\psi(y_i)\|^2$, which is used in conventional manifold learning problems, our analysis indicates that the equilibrium cannot be guaranteed. To show the existence of equilibrium for the proposed MR-GAN architecture, we use the following definition of the ϵ-approximate equilibrium.

Equilibrium ([3]). A pair of mixed strategies $(\mathcal{S}_u,\mathcal{S}_v)$ is an ϵ-approximate equilibrium if, for some value V,

$$\forall v \in \mathcal{V}, \mathop{\mathbb{E}}_{u\sim\mathcal{S}_u} F(u,v) \le V+\epsilon; \tag{4.2}$$

$$\forall u \in \mathcal{U}, \mathop{\mathbb{E}}_{v\sim\mathcal{S}_v} F(u,v) \ge V-\epsilon. \tag{4.3}$$

If the strategies $\mathcal{S}_u,\mathcal{S}_v$ are pure strategies, then this pair is called an ϵ-approximate pure equilibrium.

Theorem 4.2. *If the generator can approximate any point mass by $\mathbb{E}_{h\sim\mathcal{D}_h}[\|G_u(h)-x\|]\le\epsilon$, then there exists a universal constant $C>0$ such that for any ϵ, there exist $T=\frac{C\Delta^2 p\log(LL'L_\phi p/\epsilon)}{\epsilon^2}$ generators $G_{u1},\dots,G_{uT}$. Let $\mathcal{S}_u$ be a uniform distribution on u_i and D be a discriminator that outputs $1/2$; then $(\mathcal{S}_u, D)$ is an ϵ-approximate equilibrium for MR-GAN.*

Proof. See Appendix C. ■

Note that in the above result, the generator uses mixed strategies, which means that the generated data comes from a mixture of generators. One can add an output layer of ReLU activation functions to the generators to construct an integrated neural network of the generator, and the output is uniformly distributed over the results from the T generators in the theorem. One possible construction can be found in Lemma 4 in [3].

Stable training. From both the theoretical and the practical perspectives, the training of GANs remains a challenging problem, one of which is the issue of instability in optimizing GANs. It is presented in [31] that the training dynamics in "(stochastic) gradient descent" form of GAN optimization can be well analyzed by the method of nonlinear differential equations (ODEs), thus providing a characterization of the "stability" of GAN training. It is important to show that MR-GAN also falls into this general framework to characterize the training dynamics and to show that the proposed MR-GAN can stabilize the training process.

Assuming that the generator and discriminator networks are parameterized by the sets of parameters, u and v, respectively, we investigate the problem of analyzing stability of approaches based on stochastic gradient descent to solve (3.2). That is, we take simultaneous gradient steps in both u and v.

All our conditions are imposed on both (u^*, v^*) and all equilibrium points in a small neighborhood around it. Given the above consideration, our focus is on proving the stability of the dynamical system around equilibrium points, i.e., points θ^* for which $h(\theta^*) = 0, h(\theta) = \nabla F(\theta)$. We now discuss conditions under which we can guarantee exponential stability, which is originally defined for a dynamic system as follows.

Stability ([15]). Consider a system consisting of variables $\theta \in \mathbb{R}^n$ whose time derivative is defined by $h(\theta)$ as

$$h(\theta) = \nabla F(\theta). \tag{4.4}$$

Let $\theta(t)$ denote the state of the system at some time t. Then an equilibrium point of the system in (4.4) is

- stable if for each $\epsilon > 0$, there is $\delta = \delta(\epsilon) > 0$ such that

$$\|\theta(0)\| \leq \delta, \|\theta(t)\| \leq \epsilon \quad \forall t \geq 0; \tag{4.5}$$

- asymptotically stable if it is stable and $\delta > 0$ can be chosen such that

$$\|\theta(0)\| \leq \delta, \lim_{t\to\infty} \theta(t) = 0; \tag{4.6}$$

- exponentially stable if it is asymptotically stable and $\delta, k, \lambda > 0$ can be chosen such that

$$\|\theta(0)\| \leq \delta, \|\theta(t)\| \leq k\|\theta(0)\| \exp(-\lambda t). \tag{4.7}$$

Specifically, we invoke the well-known linearization theorem [15] analyzed for GANs training dynamics [31], which states that if the Jacobian of the dynamical system $\mathbf{J} = \partial h(\theta)/\partial\theta|\theta = \theta^*$ evaluated at an equilibrium point is Hurwitz (which has all strictly negative eigenvalues, $Re(\lambda_i(\mathbf{J})) < 0$, for all $i = 1, \ldots, n$), then the optimization of the GAN system training will converge to θ^* for some nonempty region around θ^* at an exponential rate. This means that the system is locally asymptotically stable, or more precisely, locally exponentially stable. Thus, an important contribution here is a proof of the following fact: under some conditions, the Jacobian of the dynamical system given by the proposed GAN update is a Hurwitz matrix at equilibrium. For simplicity, we denote the equilibrium point of the min-max game corresponding to GAN training by (u^*, v^*), which are the parameter sets of the discriminator and the generator at the equilibrium points. Recall that

$$\begin{aligned} F(u, v) = \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \mathop{\mathbb{E}}_{y\sim\mathcal{D}_{G_u}} &[\phi D_v(x) + \phi(1 - D_v(y)) \\ &+ \lambda\|\nabla_{\mathcal{M}}(\psi(y) - \psi(x))\|^2]. \end{aligned} \tag{4.8}$$

The gradient steps in both u and v are taken simultaneously, resulting in the following gradient differential equations:

$$\dot{u} = \nabla_u F(u, v), \quad \dot{v} = \nabla_v F(u, v). \tag{4.9}$$

Theorem 4.3. *The dynamical system defined by the MR-GAN objective in* (3.2) *and the updates in* (4.9) *is locally exponentially stable with respect to an equilibrium point* (u^*, v^*).

Proof. See Appendix D. ■

This shows that the proposed MR-GAN is locally exponentially stable. That is, for some region around an equilibrium of the updates, the gradient updates will converge to this equilibrium at an exponential rate. As an interesting note, Wasserstein GANs [2] are not even asymptotically stable [31]. However, adding the manifold regularization term makes them locally exponentially stable. Different from the analysis in [31] and the proposed GAN training architecture therein, we can guarantee the existence of the equilibrium around which the training has stable convergence. However, the analysis in [31] does not guarantee so, as it only demonstrates convergence if there do exist points that satisfy certain criteria. Thus, we provide a systematic analysis that proves the existence of the equilibrium and the stable convergence.

Optimal embedding function. The function ψ embeds the data into a low-dimensional subspace, and thus it can prevent overfitting in the training phase. However, to prevent overfitting, one can also use a smaller value of the regularizer parameter λ and employ $\psi(x) = x$ to exploit the complete geometric information in the data. Nevertheless, setting $\psi(x) = x$ is associated with highest computational complexity since no dimension reduction is introduced. Thus, a trade-off between computational complexity for training and the amount of the exploited geometric information of the data exists. On the other hand, we find in our experiments that the computational complexity is reasonable when we use $\psi(x) = x$.

For completion of the theory and to account for the scenarios where low computational complexity is desperately desired, we study the extreme case where $\psi(x)$ is a 1-dimensional embedding, i.e., $\psi : \mathbb{R}^d \to \mathbb{R}$.

Since ψ embeds the data into a 1-dimensional subspace, one can imagine that different choices of the embedding functions can lead to different qualities of the generated data by MR-GANs. Hence, it is important to find the optimal form of the embedding function ψ. As the regularized objective only takes effect in the training of the generator, we can write the joint optimization of finding the best generator and the embedding function ψ in an empirical fashion as

$$\min_{u \in \mathcal{U}, \psi} \frac{1}{m} \sum_{i=1}^{m} [\phi(1 - D_v(y_i)) + \lambda \|\nabla_{\mathcal{M}}(\psi(y_i) - \psi(x_i))\|^2]. \tag{4.10}$$

We provide the result in the following theorem for such a case.

Theorem 4.4. *The optimal one-dimensional embedding function* $\psi(x)$ *exists and admits the following representation:*

$$\psi(x) = \sum_{i=1}^{m} \alpha_i K(x_i, x), \tag{4.11}$$

where $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ *is a Mercer kernel.*

Proof. See Appendix E. ■

Here, we still need to find the coefficients α_i for the finite-dimensional space. A good method is indicated in [4]. First, one fixes the type of kernel function K and optimizes the problem (4.10) with respect to α_i. Then, the optimal α_i can be easily found using simple first-order derivative methods.

For the embedding function ψ which gives the embedding with a dimension that is higher than 1, one can use auto-encoders to encode the high-dimensional data into a low-dimensional subspace.

5. Experiments.

We corroborate our theoretical results using synthetic data and real datasets (both natural and scientific images). We present the performance comparison with some widely used benchmark GAN architectures.[5] We employ the recently proposed *geometry score* (GS) metric [17] for assessing the quality of generated samples and detecting various levels of failure models. GS compares the topological properties of the underlying real data manifold and the generated one, which provides both qualitative and quantitative means for evaluation of the results generated by GANs. It was shown in [17] that GS is more expressive in capturing various failure modes of GANs compared to its conventional counterparts, such as *inception score* [40] and *Fréchet inception distance* [11]. A lower value of GS indicates a better match between the generated data and the real data. Furthermore, both inception score and Fréchet inception distance evaluate how well the real and fake distributions are aligned in the feature space of the Inception-v3 network that is trained on the ImageNet dataset of natural images. However, these scores are meaningless in the context of scientific data because they are not natural images and are completely agnostic to the scientific characteristics of the data. As a result, GS, which is agnostic to the type of data and compares the topological characteristics, is a better metric. For completeness, we also report Fréchet inception distance in Appendix A. All the scores are empirically computed over 10,000 samples.

5.1. Synthetic data.

To illustrate the impact of the proposed regularization in the training of the generator, we train the original GAN architecture [10] (using Adam Optimizer with a learning rate of $\gamma = 1e{-}3$ for both networks) on a 2D mixture of 8 Gaussians evenly arranged in a circle. However, the circle of the Gaussian mixture lies in a hyperplane in a 3D space. We show this dataset model in the third subfigure in Figure 1. Therefore, the generator has to search for 2D submanifolds in a 3D space. The first two subfigures in Figure 1 show the GAN training results of this model after $10,000$ training iterations. We present the result of the original GAN in the first subfigure and that of MR-GAN in the second. For MR-GAN, we set the kernel scale parameter $\rho = 128$ and the regularization parameter $\lambda = 0.5$. We can clearly observe from the comparison in the figure that the original GAN misses one of the 8 modes and the problem of mode collapse happens. The proposed MR-GAN learns to evenly spread the probability mass and converges to all the 8 modes without any mode collapse. Second, we can see from the figure that the data mass generated by the proposed GAN architecture lies heavily within the mode, and the probability mass resembles the real probability in the third subfigure very well. However, the data mass generated by the original GAN scatters around the mode, compared to the result generated by MR-GAN. Furthermore, the results generated

[5] Additional results are provided in Appendix A.

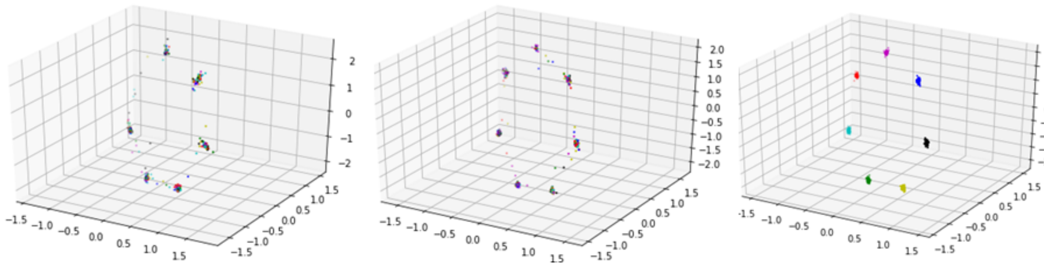


Figure 1. *MR-GANs avoid the mode collapse problem and generalize better on a toy* $2D$ *mixture of Gaussian dataset in a* $3D$ *ambient space.* (a) *Original GAN.* (b) *MR-GAN.* (c) *Ground truth.*

Table 1
GS ($\times$1e−3) *for the CelebA MR-DCGAN* ($\rho = 0.5$).

λ	0.5	0.2	0.1	0
GS ↓	44.5 ± 8.4	**37.5** ± **8.4**	38.1 ± 8.6	48.9 ± 6.2

by the original GAN have a GS of 0.909, and the results generated by the proposed MR-GAN architecture have a GS of 0.442, which is an improvement of 51.4%.

5.2. Natural image dataset.

In this section, we compare the performance of MR-GANs on a popular computer vision application with natural images—celebrity face generation.

In this experiment, we use the CelebA dataset [25], which is composed of $202,599$ images of celebrity faces. We trained a manifold regularized DCGAN (MR-DCGAN) [37] using 90% of the images from the CelebA dataset. As in the DCGAN case, we rescale the data to lie in the range $[-1,1]$. We use the same architecture as the DCGAN implementation in the discriminator and generator networks. We also use the Adam Optimizer with a learning rate of $\gamma = 2\text{e}{-4}$ for both networks. For the embedding function ψ, we use a convolutional autoencoder that embeds the training set of the CelebA dataset into a 100-dimensional latent space.

We train the network with different values of λ, ρ as explained earlier and report the quality of each GAN in Table 1. These experiments are performed over 5 independent runs. We see that adding the proposed manifold regularization significantly improves the performance of the DCGAN (shown with $\lambda = 0.0$), leading to a GS that is lower by about 23%. Samples from the MR-DCGAN are shown in Figure 2(a).

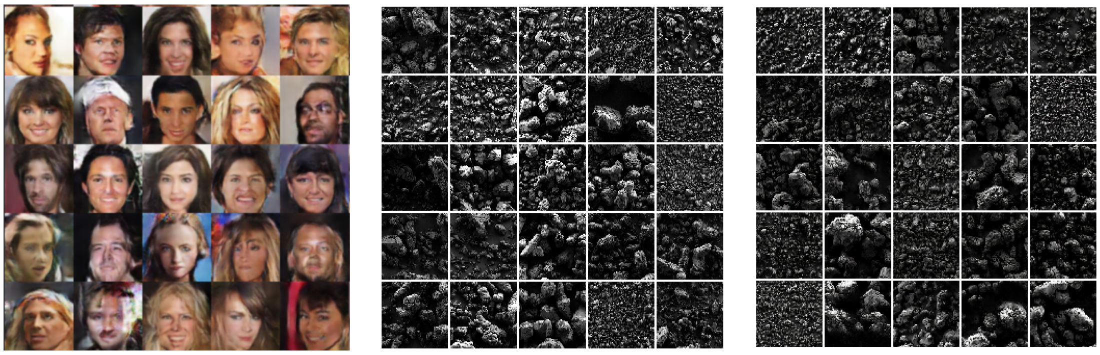
5.3. Scientific dataset.

In this experiment, we consider a scientific application, where the GAN is exploited to generate synthetic *scanning electron microscopy* (SEM) images of crystalline organic materials.

Application. SEM is an important analytical tool for nano-, meso-, and macro-scale characterization of materials that has been frequently employed in a variety of fields, including chemistry, material science, biology, and physics [9].

The SEM image dataset used in this work contains 59,690 images from 30 classes. Each class is composed of SEM images of 2,4,6-triamino-1,3,5-trinitrobenzene (TATB) crystalline samples produced with various synthesis reaction conditions.

Implementation details. We trained a MR-DCGAN using 90% (47,706 out of 59,690) of images from the whole SEM image dataset. We follow the setup introduced in section 5.2:



(a) MR-DCGAN generated celebrity images

(b) Real SEM images

(c) MR-DCGAN generated SEM images

Figure 2. *Image quality comparison for* (a) *CelebA and* (b), (c) *material science application.*

Table 2
GS (×1e−3) *for the SEM MR-DCGAN* ($\rho = 128$).

λ	0.005	0.01	0.02	0.1	0
GS ↓	55.2 ± 10.6	**42.0 ± 9.4**	62.0 ± 10.6	61.0 ± 5.6	144 ± 3.6

Table 3
GS (×1e−3) *for the SEM MR-DCGAN* ($\rho = 1280$).

λ	0.005	0.01	0.02	0.1	0
GS ↓	116 ± 26.4	98.6 ± 11.5	**78.2 ± 10.8**	129 ± 23.3	144 ± 3.6

the data is again rescaled to lie in the range $[-1, 1]$. The discriminator and generator network architectures are the same as in the DCGAN implementation. We use the Adam Optimizer with a learning rate of $\gamma = 3\text{e}{-4}$ for both networks. The embedding function ψ is the pixels of SEM images, which belong to a $64 \times 64 \times 1 = 4096$-dimensional pixel space.

Results and discussion. We train the networks with two ρ values (128 and 1280). For both of them, we examine different values of λ and report the GAN quality in Table 2 and Table 3. These experiments are performed over 5 independent runs. We observe that the MR-DCGAN can significantly outperform the baseline DCGAN ($\lambda = 0$), with a best-case GS improvement of $\sim 70\%$ when $\rho = 128$ and $\sim 45\%$ when $\rho = 1280$. This corroborates our hypothesis that by incorporating intrinsic manifold information in order to accurately capture the complex geometries of scientific datasets will results in significant gains. Sample results from the MR-DCGAN are shown in Figure 2(b) and (c).

6. Conclusion.

We studied the problem of training GANs. We proposed a manifold regularization method to force the generator to respect the unique manifold geometry of the real data in order to generate high quality data. Furthermore, we theoretically proved that the incorporation of this regularization term in any class of GANs leads to improved performance. We empirically showed that by incorporating intrinsic manifold information in order to accurately capture the complex geometries, the proposed manifold regularization helps in avoiding mode collapse and leads to stable training on both natural and scientific datasets. There are still many interesting questions that remain to be explored in the future such as establishing the global convergence properties of GAN training. It will also be interesting to explore the connection between the proposed method and the recently proposed Jacobian clamping method [34]. Other cases where both the discriminator and the generator are regularized or there is noise present in the training data and further experiments with state-of-the-art GAN architectures may also be interesting to investigate. Finally, application of MR-GAN on other scientific applications is expected to produce similar gains and is a worthwhile future direction.

Appendix A. Additional results.

A.1. MNIST dataset. We also test our approach on the MNIST dataset [19] of handwritten digits. We compare the proposed GAN architecture based on the recently proposed model, i.e., Wasserstein GAN (WGAN) [2]. We use the RMSProp Optimizer with a learning rate of $\gamma = 1e - 4$ for both networks. In the following tables, we quantify the performance in terms of the GS for the proposed MR-WGAN architecture with different values of kernel scale parameter ρ and with different values of regularization parameter λ after 300K training iterations.

First, we set the kernel scale parameter ρ to 6.4 and vary the value of the regularization parameter λ from 0.05 to 0.4, as shown in Table 4. Note that WGAN yields GS of 0.414, which is shown with $\lambda = 0$. When the value of λ is small, we observe the improvement in GS for manifold regularized (MR-WGAN) compared to the results of WGAN. When $\lambda = 0.2$, the proposed MR-WGAN has GS = 0.384, which provides an improvement of 7.25% in GS. We also provide various GS results when $\rho = 10$ in Table 5. When λ is small, we again observe improvement. When $\lambda = 0.01$, we have the best result, and the GS of the results generated by MR-GAN is 0.372, which is an improvement of 10.14%.

In Figure 3, we present the result for the WGAN and the proposed MR-WGAN for the MNIST dataset. The results are obtained after 10,000 training iterations. We can see that the proposed GAN architecture achieves better results.

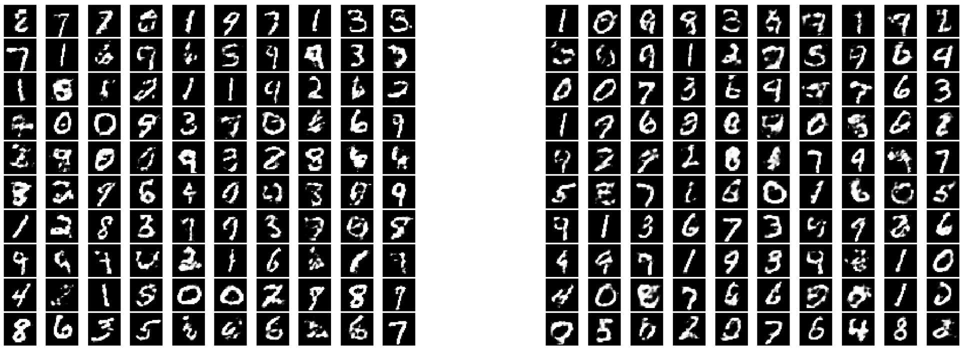
We also report the inception score and Fréchet inception distance (FID) for the generated results for different architectures in Table 6. The MR-WGAN model used in the table is the one that generated the best GS in previous experiments, i.e., $\rho = 10$ and $\gamma = 0.01$. We can observe from the table that when our proposed manifold regularizer is employed to the WGAN architecture, the performance in terms of either inception score or FID is improved.

Table 4
GS for the MNIST MR-WGAN ($\rho = 6.4$).

λ	0.05	0.1	0.2	0.3	0.4	0
GS	0.405	0.403	**0.384**	0.444	0.441	*0.414*

Table 5
GS for the MNIST MR-WGAN ($\rho = 10$).

λ	0.005	0.01	0.02	0.1	0.2	0
GS	0.382	**0.372**	0.379	0.404	0.506	*0.414*



(a) Randomly generated digits from the WGAN (b) Randomly generated digits from the MR-GAN

Figure 3. *Performance comparison for digits generation.*

Table 6

Inception score and FID.

	WGAN	MR-WGAN
Inception score	2.02	2.14
FID	67.88	59.69

Table 7

GS ($\times$1e$-$3) *and FID score for the CelebA MR-DCGAN* ($\rho = 0.5$)*.*

λ	0.5	0.2	0.1	0
GS $\downarrow$	44.5 ± 8.4	$\mathbf{37.5 \pm 8.4}$	38.1 ± 8.6	48.9 ± 6.2
FID $\downarrow$	51.3 ± 3.3	$\mathbf{50.5 \pm 1.3}$	54.9 ± 3.4	51.0 ± 3.7

Specifically, the inception score increases from 2.02 to 2.14, and the FID decreases from 67.88 to 59.69.

A.2. CelebA dataset. In Table 7, we report the GS scores and the FID scores averaged over 5 independent experimental runs. As can be seen from the table, the manifold regularizer improves the performance in terms of both of these metrics.

A.3. SEM dataset. For the scientific application, we calculate the FID-like score for the real and generated images using a trained Wide Resnet for a relevant classification problem (see [44] for more details) with high accuracy (92.3%). Specifically, the 64-dimensional last-pooling layer of the 30-class classification model is used to capture the features of input images. As can be seen from Tables 8 and 9, the manifold regularizer improves the performance in terms of both the GS score as well as the FID score. The scores are calculated over 5 independent runs, and the mean and the standard deviation are reported.

Appendix B. Proof of Theorem 4.1.

Theorem B.1. *Let* $\hat{\mathcal{D}}_{real}$ *and* $\hat{\mathcal{D}}_{G_u}$ *be empirical versions with at least* m *samples each for MR-GAN. There is a universal constant* C *such that when* $m \geq \frac{Cp\log(LL_\phi p/\epsilon)(\Delta+4\lambda M^2)^2}{\epsilon^2}$*, we have, with probability at least* $1-\exp(1-p)$*,*

Table 8

GS ($\times$1e$-$3) *and FID-like score for the SEM MR-DCGAN* ($\rho = 128$)*.*

λ	0.005	0.01	0.02	0.1	0
GS ↓	55.2 ± 10.6	**42.0 ± 9.4**	62.0 ± 10.6	61.0 ± 5.6	144 ± 3.6
FID ↓	52.0 ± 2.0	**46.0 ± 3.2**	48.0 ± 2.9	53.3 ± 3.7	54.4 ± 2.9

Table 9

GS ($\times$1e$-$3) *and FID-like score for the SEM MR-DCGAN* ($\rho = 1280$)*.*

λ	0.005	0.01	0.02	0.1	0
GS ↓	116 ± 26.4	98.6 ± 11.5	**78.2 ± 10.8**	129 ± 23.3	144 ± 3.6
FID ↓	61.6 ± 4.2	**51.9 ± 0.5**	57.8 ± 4.0	56.7 ± 2.0	54.4 ± 2.9

$$|F(u,v) - \hat{F}(u,v)| \leq \epsilon. \tag{B.1}$$

Proof. Let $\mathcal{X}$ be a finite set such that every point in $\mathcal{V}$ is within distance $\epsilon/8LL_\phi$ of a point in X (a so-called $\epsilon/8LL_\phi$-net). Standard constructions give an X satisfying $\log|\mathcal{X}| \leq O(p\log(LL_\phi p/\epsilon))$. For every $v \in \mathcal{X}$, by Hoeffding's inequality we know that

$$\begin{aligned}
&\Pr\left(|f(\mathcal{D}_{real}, \mathcal{D}_{G_u}, v) - f(\hat{\mathcal{D}}_{real}, \hat{\mathcal{D}}_{G_u}, v)| \geq \frac{\epsilon}{4}\right)\\
&\leq 2\exp\left(-\frac{m^2\frac{\epsilon^2}{16}}{m(2\Delta + 8\lambda M^2)^2}\right)\\
&= 2\exp\left(-\frac{m\epsilon^2}{32(\Delta + 4\lambda M^2)^2}\right), \qquad \text{(B.2)}
\end{aligned}$$

where $f(\mathcal{D}_{real}, \mathcal{D}_{G_u}, v) = \mathbb{E}_{x\sim\mathcal{D}_{real}}\mathbb{E}_{y\sim\mathcal{D}_{G_u}}\phi(1 - D_v(y)) + \lambda\|\nabla_{\mathcal{M}}(\psi(y) - \psi(x)\|^2$. Thus, we can union bound over all $v \in \mathcal{X}$, for large enough constant C,

$$\begin{aligned}
&\Pr\left(|f(\mathcal{D}_{real}, \mathcal{D}_{G_u}, v) - f(\hat{\mathcal{D}}_{real}, \hat{\mathcal{D}}_{G_u}, v)| \geq \frac{\epsilon}{4}\right)\\
&\leq 2|\mathcal{X}|\exp\left(-\frac{m\epsilon^2}{32(\Delta + 4\lambda M^2)^2}\right) \qquad \text{(B.3)}\\
&= \exp\left(\log 2|\mathcal{X}| - \frac{m\epsilon^2}{32(\Delta + 4\lambda M^2)^2}\right) \qquad \text{(B.4)}\\
&\leq \exp\left(Cp\log(LL_\phi p/\epsilon) - \frac{m\epsilon^2}{32(\Delta + 4\lambda M^2)^2}\right). \qquad \text{(B.5)}
\end{aligned}$$

Choose m such that $m \geq \frac{Cp\log(LL_\phi p/\epsilon)(\Delta + 4\lambda M^2)^2}{\epsilon^2}$, and thus, with high probability (at leat $1 - \exp(-p)$) we have $|f(\mathcal{D}_{real}, \mathcal{D}_{G_u}, v) - f(\hat{\mathcal{D}}_{real}, \hat{\mathcal{D}}_{G_u}, v)| \leq \frac{\epsilon}{4}$.

Now, for $v \in \mathcal{V}$ and $v' \in \mathcal{X}$ such that $\|v - v'\| \leq \epsilon/8LL_\phi$, we have

$$\begin{aligned}
&|f(\mathcal{D}_{real}, \mathcal{D}_{G_u}, v) - f(\hat{\mathcal{D}}_{real}, \hat{\mathcal{D}}_{G_u}, v)|\\
\leq&|f(\mathcal{D}_{real}, \mathcal{D}_{G_u}, v') - f(\hat{\mathcal{D}}_{real}, \hat{\mathcal{D}}_{G_u}, v')| \qquad \text{(B.6)}\\
&+ |f(\mathcal{D}_{real}, \mathcal{D}_{G_u}, v') - f(\mathcal{D}_{real}, \mathcal{D}_{G_u}, v)| \qquad \text{(B.7)}
\end{aligned}$$

$$+ |f(\hat{\mathcal{D}}_{real}, \hat{\mathcal{D}}_{G_u}, v') - f(\hat{\mathcal{D}}_{real}, \hat{\mathcal{D}}_{G_u}, v)| \tag{B.8}$$

$$\leq \epsilon/4 + \epsilon/8 + \epsilon/8 = \epsilon/2. \tag{B.9}$$

The value of $\epsilon/8$ results from Lipschitz continuity.

Similarly, we can bound

$$| \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \phi\left(D_v(x)\right) - \mathop{\mathbb{E}}_{x\sim\hat{\mathcal{D}}_{real}} \phi\left(D_v(x)\right) | \leq \epsilon/2 \tag{B.10}$$

with $m \geq \frac{Cp\log(LL_\phi p/\epsilon)\Delta^2}{\epsilon^2}$.

Since $F(u, v) = f(\mathcal{D}_{real}, \mathcal{D}_{G_u}, v) + \mathbb{E}_{x\sim\mathcal{D}_{real}} \phi\left(D_v(x)\right)$, choose m such that

$$m \geq \frac{Cp\log(LL_\phi p/\epsilon)(\Delta + 4\lambda M^2)^2}{\epsilon^2},$$

and we have the desired result. ■

Appendix C. Proof of Theorem 4.2.

Theorem C.1. *If the generator can approximate any point mass by $\mathbb{E}_{h\sim\mathcal{D}_h}[\|G_u(h) - x\|] \leq \epsilon$, there is a universal constant $C > 0$ such that for any ϵ, there exist $T = \frac{C\Delta^2 p\log(LL'L_\phi p/\epsilon)}{\epsilon^2}$ generators $G_{u1}, \dots, G_{uT}$. Let $\mathcal{S}_u$ be a uniform distribution on u_i, and D is a discriminator that outputs $1/2$; then $(\mathcal{S}_u, D)$ is an ϵ-approximate equilibrium for MR-GAN.*

Proof. We first prove the value of the function $F(u, v)$ of the game at the equilibrium must be equal to $2\phi(1/2)$. This strategy has payoff $2\phi(1/2)$ no matter what the generator does, so $V \geq 2\phi(1/2)$.

For the generator, we use the assumption that for any point x and any $\epsilon > 0$, there is a generator (which we denote by $G_{x,\epsilon}$) such that $\mathbb{E}_{h\sim\mathcal{D}_h}\|G_{x,\epsilon}(h) - x\| \leq \epsilon$. Now for any $\alpha > 0$, consider the following mixture of generators: sample $x \sim \mathcal{D}_{real}$, then use the generator $G_{x,\alpha}$. Let $\mathcal{D}_\alpha$ be the distribution generated by this mixture of generators. The Wasserstein distance between $\mathcal{D}_\alpha$ and $\mathcal{D}_{real}$ is bounded by α. Since the discriminator is L'-Lipschitz, it cannot distinguish between $\mathcal{D}_\alpha$ and $\mathcal{D}_{real}$. In particular we know for any discriminator D_v that

$$| \mathop{\mathbb{E}}_{y\sim\mathcal{D}_\alpha} [\phi(1 - D_v(y))] - \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} [\phi(1 - D_v(x))]| \leq O(L_\phi L'\alpha). \tag{C.1}$$

Therefore,

$$\begin{aligned}
&\max_{v\in\mathcal{V}} \mathop{\mathbb{E}}_{y\sim\mathcal{D}_\alpha} \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} [\phi(D_v(x))] + [\phi(1 - D_v(y))] + \lambda\|\nabla_{\mathcal{M}}(\psi(y) - \psi(x))\|^2 \\
&\quad \leq \max_{v\in\mathcal{V}} \mathop{\mathbb{E}}_{y\sim\mathcal{D}_\alpha} \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} -\phi(1 - D_v(x)) + \phi(1 - D_v(y)) \\
&\qquad + \lambda\|\nabla_{\mathcal{M}}(\psi(y) - \psi(x))\|^2 + \max_{v\in\mathcal{V}} \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \phi(D_v(x)) + \phi(1 - D_v(x)) \\
&\quad \leq O(L_\phi L'\alpha) + \lambda L_\psi^2 \alpha^2 + 2\phi(1/2).
\end{aligned} \tag{C.2}$$

Here the last step uses the assumption that ϕ is concave. Therefore, the value is upper-bounded by $V \leq O(L_\phi L'\alpha) + \lambda L_\psi^2\alpha^2 + 2\phi(1/2)$ for any α. Taking limit of α to 0, we have $V = 2\phi(1/2)$.

The value of the game is $2\phi(1/2)$ in particular means the optimal discriminator cannot do anything other than a random guess. Therefore, we will use a discriminator that outputs $1/2$. Next we will construct the generator.

Let $\{\mathcal{S}'_u, \mathcal{S}'_v\}$ be the pair of optimal mixed strategies as in the theorem and V be the optimal value. We will show that randomly sampling T generators from $\mathcal{S}'_u$ gives the desired mixture with high probability.

Construct $\epsilon/4LL'L_\phi$-nets $\mathcal{V}$ for the parameters of the discriminator (for any $v, v' \in \mathcal{V}$, $\|v - v'\| \leq \epsilon/4LL'L_\phi$). By standard construction, the sizes of these ϵ-nets satisfy $\log|\mathcal{V}| \leq C'p\log(LL'L_\phi p/\epsilon)$ for some constant C'. Let $u_1, \ldots, u_T$ be independent samples from $\mathcal{S}'_u$. By Hoeffding's inequality, for any $v \in \mathcal{V}$, we know that

$$\begin{aligned} P\left(\mathop{\mathbb{E}}_{i\in[T]} F(u_i, v) - \mathop{\mathbb{E}}_{u\in\mathcal{U}} F(u, v) \geq \frac{\epsilon}{2}\right) &\leq \exp\left(-\frac{2T^2\frac{\epsilon^2}{4}}{T4\Delta^2}\right) \\ &= \exp\left(-\frac{T\epsilon^2}{8\Delta^2}\right). \end{aligned} \tag{C.3}$$

Now for all $v \in \mathcal{V}$, with union bound, we have

$$\begin{aligned} \forall v \in \mathcal{V}, \quad & P\left(\mathop{\mathbb{E}}_{i\in[T]} F(u_i, v) - \mathop{\mathbb{E}}_{u\in\mathcal{U}} F(u, v) \geq \frac{\epsilon}{2}\right) \\ & \leq |\mathcal{V}| \exp\left(-\frac{T\epsilon^2}{8\Delta^2}\right) \\ & \leq \exp\left(C'p\log(LL'L_\phi p/\epsilon) - \frac{T\epsilon^2}{8\Delta^2}\right). \end{aligned} \tag{C.4}$$

Thus, when $T = \frac{C\Delta^2 p\log(LL'L_\phi p/\epsilon)}{\epsilon^2}$ and $C \geq 8C'$, with high probability,

$$\mathop{\mathbb{E}}_{i\in[T]} F(u_i, v) \leq \mathop{\mathbb{E}}_{u\in\mathcal{U}} F(u, v) + \frac{\epsilon}{2}. \tag{C.5}$$

By construction of the net, we have $\|v - v'\| \leq \frac{\epsilon}{4LL'L_\phi}$. It is easy to find that $F(u, v)$ is $2LL'L_\phi$-Lipschitz with respect to v, and therefore,

$$\begin{aligned} \mathop{\mathbb{E}}_{i\in[T]} F(u_i, v') &\leq \mathop{\mathbb{E}}_{i\in[T]} F(u_i, v) + 2LL'L_\phi \frac{\epsilon}{4LL'L_\phi} \\ &= \mathop{\mathbb{E}}_{i\in[T]} F(u_i, v) + \frac{\epsilon}{2}. \end{aligned} \tag{C.6}$$

Together with the inequality (C.5), we obtain

$$\forall v' \in \mathcal{V}, \mathop{\mathbb{E}}_{i\in[T]} F(u_i, v') \leq 2\phi(1/2) + \epsilon. \tag{C.7}$$

This means the mixture of generators can win against any discriminator. By probabilistic argument, we know there must exist such generators. The discriminator (outputs $1/2$) obviously achieves value V no matter what the generator is. Therefore, we get an approximate equilibrium. ■

Appendix D. Proof of Theorem 4.3.

Theorem D.1. *The dynamical system defined by the MR-GAN objective in* (3.2) *and the updates in* (4.9) *is locally exponentially stable with respect to an equilibrium point* (u^*, v^*).

Proof. To derive the Jacobian, we begin with subtly different algebraic form of the GAN objective by

$$\begin{aligned} F(u,v) = \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \int_{\mathcal{Y}} &(p_u(y)[\phi D_v(x) + \phi(1 - D_v(y))] \\ &+ \lambda\|\nabla_{\mathcal{M}}(\psi(y) - \psi(x))\|^2)dy. \end{aligned} \tag{D.1}$$

Thus, we have the following form of the dynamic ODE system:

$$\begin{aligned} \dot{u} &= -\frac{\partial F(u,v)}{\partial u} \\ &= -\mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \int_{\mathcal{Y}} (\nabla_u p_u(y)\phi(1 - D_v(y)) \\ &\qquad + \lambda\|\nabla_{\mathcal{M}}(\psi(y) - \psi(x))\|^2)dy \end{aligned} \tag{D.2}$$

$$\begin{aligned} \dot{v} &= \frac{\partial F(u,v)}{\partial v} \\ &= \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \mathop{\mathbb{E}}_{y\sim p_u(y)} [\phi' D_v(x)\nabla_v D_v(x) \\ &\qquad - \phi'(1 - D_v(y))\nabla_v D_v(y)]. \end{aligned} \tag{D.3}$$

The Jacobian matrix $\mathbf{J}$ consists of blocks as

$$\mathbf{J} = \begin{pmatrix} \mathbf{J}_{vv} & \mathbf{J}_{vu} \\ \mathbf{J}_{uv} & \mathbf{J}_{uu}, \end{pmatrix}. \tag{D.4}$$

Then $\mathbf{J}_{vv}$ is

$$\mathbf{J}_{vv} = \nabla_v^2 F(u,v)\Big|_{\substack{u=u^* \\ v=v^*}} = \frac{\partial \dot{v}}{\partial v}\Big|_{\substack{u=u^* \\ v=v^*}} = \frac{\partial \dot{v}|_{u=u^*}}{\partial v}\Big|_{v=v^*} \tag{D.5}$$

$$= \frac{\partial(\mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} [\phi' D_v(x)\nabla_v D_v(x)}{\partial v}\Bigg|_{v=v^*} \tag{D.6}$$

$$- \frac{\partial(\mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} [\phi'(1 - D_v(x))\nabla_v D_v(x)])}{\partial v}\Bigg|_{v=v^*} \tag{D.7}$$

$$= \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \phi''(D_v(x))\nabla_v D_v(x)\nabla_v^T D_v(x)\Big|_{v=v^*} \tag{D.8}$$

$$+ \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \phi'(D_v(x))\nabla_v^2 D_v(x)\Big|_{v=v^*} \tag{D.9}$$

$$+ \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \phi''(D_v(x))\nabla_v D_v(x)\nabla_v^T D_v(x)\Big|_{v=v^*} \tag{D.10}$$

$$- \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \phi'(D_v(x))\nabla_v^2 D_v(x)\Big|_{v=v^*} \tag{D.11}$$

$$= 2\phi''\left(\frac{1}{2}\right) \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \nabla_v D_v(x)\nabla_v^T D_v(x)\Big|_{v=v^*}. \tag{D.12}$$

The matrix $\mathbf{J}_{vu}$ is

$$\mathbf{J}_{vu} = \frac{\partial \dot{v}}{\partial u}\bigg|_{u=u^*,v=v^*} = \frac{\partial \dot{v}|_{v=v^*}}{\partial u}\bigg|_{u=u^*} \tag{D.13}$$

$$= \frac{\partial}{\partial u} \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \mathop{\mathbb{E}}_{y\sim p_u(y)} \left[-\phi'\left(\frac{1}{2}\right)\nabla_v D_v(y)\right]\bigg|_{u=u^*,v=v^*} \tag{D.14}$$

$$= -\phi'\left(\frac{1}{2}\right) \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \int_{\mathcal{Y}} \nabla_v D_v(y)\nabla_u^T p_u(y)dy\bigg|_{u=u^*,v=v^*}. \tag{D.15}$$

The matrix $\mathbf{J}_{uv}$ is

$$\mathbf{J}_{uv} = \frac{\partial \dot{u}}{\partial v}\bigg|_{u=u^*,v=v^*} = \frac{\partial \dot{u}|_{u=u^*}}{\partial v}\bigg|_{v=v^*} \tag{D.16}$$

$$= -\frac{\partial}{\partial v} \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \int_{\mathcal{Y}} \nabla_u p_u(y)[\phi(1-D_v(x))]dy\bigg|_{u=u^*,v=v^*} \tag{D.17}$$

$$= -\mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \int_{\mathcal{Y}} \nabla_u p_u(y)(-\phi'(1-D_v(x)))\nabla_v^T D_v(x)dy\bigg|_{\substack{u=u^*\\ v=v^*}} \tag{D.18}$$

$$= \phi'(\frac{1}{2}) \mathop{\mathbb{E}}_{x\sim\mathcal{D}_{real}} \int_{\mathcal{Y}} \nabla_u p_u(y)\nabla_v^T D_v(x)dy\bigg|_{u=u^*,v=v^*} = -\mathbf{J}_{vu}. \tag{D.19}$$

Now, to show that $\mathbf{J}_{uu}$ is zero, we take any vector $\mathbf{v}$ that is a perturbation in the generator space and show that $\mathbf{v}^T\mathbf{J}_{uu} = 0$. Here, we will use the limit definition of the derivative along a particular direction $\mathbf{v}$:

$$\mathbf{v}^T\frac{\partial \dot{u}}{\partial u}\bigg|_{u=u^*,v=v^*} = \mathbf{v}^T\frac{\partial \dot{u}|\, v=v^*}{\partial u}\bigg|_{u=u^*} \tag{D.20}$$

$$= -\lim_{\substack{u-u^*=\epsilon\mathbf{v},\\ \epsilon\to 0}} \frac{\mathbb{E}_{\mathcal{D}_{real}}\int_{\mathcal{Y}}(\nabla_u p_u(y)\phi(1-D_{v*}(y)))dy}{\epsilon} \tag{D.21}$$

$$-\lim_{\substack{u-u^*=\epsilon\mathbf{v},\\ \epsilon\to 0}} \frac{\mathbb{E}_{\mathcal{D}_{real}}\int_{\mathcal{Y}}(\lambda\|\nabla_{\mathcal{M}}(\psi(y)-\psi(x))\|^2)dy}{\epsilon} \tag{D.22}$$

$$= -\lim_{\substack{u-u^*=\epsilon\mathbf{v},\\ \epsilon\to 0}} \frac{\int_{supp(\mathcal{D}_{real})}(\nabla_u p_u(y)\phi(1-D_{v*}(x))dy}{\epsilon} \tag{D.23}$$

$$= -\phi\left(\frac{1}{2}\right) \lim_{\substack{u-u^*=\epsilon\mathbf{v}\\ \epsilon\to 0}} \frac{\nabla_u \int_{supp(\mathcal{D}_{real})} p_u(y)dy}{\epsilon} = 0. \tag{D.24}$$

According to Lemma C.3 and Lemma G.2 in [31], the Jacobian matrix is full rank and Hurwitz, and the training is locally exponentially stable. ■

Appendix E. Proof of Theorem 4.4.

Theorem E.1. *The optimal one-dimensional embedding function $\psi(x)$ exists and admits the following representation:*

$$\psi(x) = \sum_{i=1}^{m} \alpha_i K(x_i, x), \tag{E.1}$$

where $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a Mercer kernel.

Proof. Consider samples y_i and x_i (same index i), $i = 1, \dots, m$, are drawn from the generator and the real dataset; we use the empirical expressions. To find the optimal embedding function ψ, we write the cost function for the generator:

$$\begin{aligned} \min_{u\in\mathcal{U},\psi} \frac{1}{m} \sum_{x\sim\mathcal{D}_{real}, y\sim\mathcal{D}_{G_u}} & [\phi(1 - D_v(y_i)) \\ & + \lambda\|\nabla_{\mathcal{M}}(\psi(y_i) - \psi(x_i))\|^2]. \end{aligned} \tag{E.2}$$

By using the samples, any function ψ derived can be uniquely decomposed into a component $\psi_{\|}$ in the linear space spanned by the kernel functions $\{K(x_i, \cdot)\}_{i=1}^m$ and a component $\psi_\perp$ orthogonal to it. Thus,

$$\psi = \psi_{\|} + \psi_\perp = \sum_{i=1}^{m} \alpha_i K(x_i, \cdot) + \psi_\perp. \tag{E.3}$$

By the reproducing property, the evaluation of ψ on any data point x_j is independent of the orthogonal component $\psi_\perp$:

$$\begin{aligned} \psi(x_j) &= \langle f, K(x_j, \cdot)\rangle \\ &= \left\langle \sum_{i=1}^{m} \alpha_i K(x_i, \cdot), K(x_j, \cdot) \right\rangle + \langle \psi_\perp, K(x_j, \cdot)\rangle. \end{aligned} \tag{E.4}$$

Since the second term zeros out and $\langle (x_i, \cdot), K(x_j, \cdot)\rangle = K(x_i, x_j)$, it follows that $\psi(x_j) = \sum_{i=1}^{m} \alpha_i K(x_i, x_j)$.

Indeed, we find that

$$FLF^T = \langle \psi(y), L\psi(y)\rangle - 2\langle \psi(y), L\psi(x)\rangle + \langle \psi(x), L\psi(x)\rangle, \tag{E.5}$$

where $F = [f_1, f_2, \dots, f_m]$ and $f_i = \psi(y_i) - \psi(x_i)$, $\psi(x) = [\psi(x_1), \psi(x_2), \dots, \psi(x_m)]$ and $\psi(y) = [\psi(y_1), \psi(y_2), \dots, \psi(y_m)]$. Hence, it can be further written with respect to L-norm as

$$\begin{aligned} FLF^T = & \sum_{j=1}^{m} \left\| \sum_{i=1}^{m} \alpha_i K(x_i, y_j) \right\|_L^2 + \sum_{j=1}^{m} \|\psi_\perp(y_j)\|_L^2 \\ & + \sum_{j=1}^{m} \left\| \sum_{i=1}^{m} \alpha_i K(x_i, x_j) \right\|_L^2 - \langle (\psi_{\|}(x), L\psi_\perp(y)\rangle, \end{aligned} \tag{E.6}$$

where $\psi_{\|}(x) = [\psi_{\|}(x_1), \psi_{\|}(x_2), \ldots, \psi_{\|}(x_m)]$ and $\psi_{\|}(y) = [\psi_{\|}(y_1), \psi_{\|}(y_2), \ldots, \psi_{\|}(y_m)]$. It follows that the optimal embedding function ψ of problem E.2 must have $\psi_{\perp} = 0$. Therefore, it admits a representation

$$\psi(x) = \sum_{i=1}^{m} \alpha_i K(x_i, x). \tag{E.7}$$

$\blacksquare$

Appendix F. Python code.

In this section, we provide a concise Python implementation for the manifold regularizer, with the choice of pixel-space embedding function $\psi(x)$ and the Gaussian heat kernel weight ω_{ij} as given in (2.1).

```python
import numpy as np
import torch
'''
The function kernel_product provides the Gaussian heat kernel weight wij for pixel-space embedding (x and y are two images, rho is the Gaussian heat kernel coefficient).
'''

def kernel_product(x,y,rho):
    with torch.no_grad():
        xmy = torch.sum((x-y)**2,(1,2,3))
    K = torch.exp(-xmy/rho)
    return torch.t(K)

'''
The following code snippet provides the manifold regularizer described in (10), which can be readily added to the conventional generator loss function during the backpropagation training process.

real: a batch of real images with batch size $n_b$
fake: a batch of synthetic images with batch size $n_b$, produced by the generator
'''
Wij = kernel_product(real[:int(nb/2),:],real[int(nb/2):,:], rho)

latentloss = torch.sum((fake[:int(nb/2),:]-real[:int(nb/2),:]-fake[int(nb/2):,:]+real[int(nb/2):,:])**2,(1,2,3))

mrloss = lambda*torch.mean(Wij*latentloss)
```

REFERENCES

[1] M. Arjovsky and L. Bottou, *Towards Principled Methods for Training Generative Adversarial Networks*, arXiv preprint, https://arxiv.org/abs/1701.04862, 2017.

[2] M. Arjovsky, S. Chintala, and L. Bottou, *Wasserstein generative adversarial networks*, in Proceedings of the International Conference on Machine Learning, 2017, pp. 214–223.

[3] S. Arora, R. Ge, Y. Liang, T. Ma, and Y. Zhang, *Generalization and equilibrium in generative adversarial nets (GANS)*, in Proceedings of the International Conference on Machine Learning, 2017, pp. 224–232.

[4] M. Belkin, P. Niyogi, and V. Sindhwani, *Manifold regularization: A geometric framework for learning from labeled and unlabeled examples*, J. Mach. Learn. Res., 7 (2006), pp. 2399–2434.

[5] T. Che, Y. Li, A. P. Jacob, Y. Bengio, and W. Li, *Mode Regularized Generative Adversarial Networks*, arXiv preprint, https://arxiv.org/abs/1612.02136, 2016.

[6] L. de Oliveira, M. Paganini, and B. Nachman, *Learning particle physics by example: Location-aware generative adversarial networks for physics synthesis*, Comput. Software Big Sci., 1 (2017), 4.

[7] A. Esteva, A. Robicquet, B. Ramsundar, V. Kuleshov, M. DePristo, K. Chou, C. Cui, G. Corrado, S. Thrun, and J. Dean, *A guide to deep learning in healthcare*, Nat. Med., 25 (2019), pp. 24–29.

[8] M. Frid-Adar, I. Diamant, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, *GAN-based synthetic medical image augmentation for increased cnn performance in liver lesion classification*, Neurocomputing, 321 (2018), pp. 321–331.

[9] J. I. Goldstein, D. E. Newbury, J. R. Michael, N. W. Ritchie, J. H. J. Scott, and D. C. Joy, *Scanning Electron Microscopy and X-Ray Microanalysis*, Springer, Cham, 2017.

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial nets*, in Advances in Neural Information Processing Systems, 2014, pp. 2672–2680.

[11] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, *GANs trained by a two time-scale update rule converge to a local nash equilibrium*, in Advances in Neural Information Processing Systems, 2017, pp. 6626–6637.

[12] K. Ji and Y. Liang, *Minimax Estimation of Neural Net Distance*, arXiv preprint, https://arxiv.org/abs/1811.01054, 2018.

[13] K. Ji, Y. Zhou, and Y. Liang, *Understanding estimation and generalization error of generative adversarial networks*, IEEE Trans. Inform. Theory, 67 (2021), pp. 3114–3129.

[14] B. Kailkhura, B. Gallagher, S. Kim, A. Hiszpanski, and T. Y.-J. Han, *Reliable and explainable machine-learning methods for accelerated material discovery*, NPJ Comput. Mater., 5 (2019), pp. 1–9.

[15] H. K. Khalil, *Nonlinear Systems*, 2nd ed., Prentice-Hall, Englewood Cliffs, New Jersey, 1996, pp. 5–1.

[16] M. Khayatkhoei, M. Singh, and A. Elgammal, *Disconnected Manifold Learning for Generative Adversarial Networks*, arXiv preprint, https://arxiv.org/abs/1806.00880, 2018.

[17] V. Khrulkov and I. Oseledets, *Geometry score: A method for comparing generative adversarial networks*, in Proceedings of the International Conference on Machine Learning, 2018.

[18] B. Lecouat, C.-S. Foo, H. Zenati, and V. R. Chandrasekhar, *Semi-Supervised Learning with GANs: Revisiting Manifold Regularization*, arXiv preprint, https://arxiv.org/abs/1805.08957, 2018.

[19] Y. LeCun and C. Cortes, *MNIST Handwritten Digit Database*, 2010, http://yann.lecun.com/exdb/mnist/.

[20] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, *Photo-realistic single image super-resolution using a generative adversarial network*, in Proceedings of the IEEE Conference on Computer Vision and pattern recognition, 2017.

[21] M. Lee and J. Seok, *Regularization Methods for Generative Adversarial Networks: An Overview of Recent Studies*, arXiv preprint, https://arxiv.org/abs/2005.09165, 2020.

[22] C.-L. Li, W.-C. Chang, Y. Cheng, Y. Yang, and B. Póczos, *MMD GAN: Towards Deeper Understanding of Moment Matching Network*, arXiv preprint, https://arxiv.org/abs/1705.08584, 2017.

[23] S. Liu, B. Kailkhura, D. Loveland, and Y. Han, *Generative Counterfactual Introspection for Explainable Deep Learning*, arXiv preprint, https://arxiv.org/abs/1907.03077, 2019.

[24] S. Liu, B. Kailkhura, J. Zhang, A. M. Hiszpanski, E. Robertson, D. Loveland, and T. Han, *Explainable Deep Learning for Uncovering Actionable Scientific Insights for Materials Discovery and Design*, arXiv preprint, https://arxiv.org/abs/2007.08631, 2020.

[25] Z. Liu, P. Luo, X. Wang, and X. Tang, *Deep learning face attributes in the wild*, in Proceedings of the International Conference on Computer Vision (ICCV), 2015.

[26] L. Mescheder, S. Nowozin, and A. Geiger, *The numerics of GANs*, in Advances in Neural Information Processing Systems, 2017, pp. 1825–1835.

[27] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein, *Unrolled Generative Adversarial Networks*, arXiv preprint, https://arxiv.org/abs/1611.02163, 2016.

[28] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, *Spectral Normalization for Generative Adversarial Networks*, arXiv preprint, https://arxiv.org/abs/1802.05957, 2018.

[29] Y. Mroueh, T. Sercu, and V. Goel, *MCGAN: Mean and covariance feature matching GAN*, in Proceeding of the International Conference on Machine Learning, 2017, pp. 2527–2535.

[30] M. Mustafa, D. Bard, W. Bhimji, Z. Lukić, R. Al-Rfou, and J. M. Kratochvil, *Cosmogan: Creating high-fidelity weak lensing convergence maps using generative adversarial networks*, Comput. Astrophys. Cosmol., 6 (2019), p. 1.

[31] V. Nagarajan and J. Z. Kolter, *Gradient descent GAN optimization is locally stable*, in Advances in Neural Information Processing Systems, 2017, pp. 5591–5600.

[32] H. Narayanan and S. Mitter, *Sample complexity of testing the manifold hypothesis*, in Advances in Neural Information Processing Systems, 2010, pp. 1786–1794.

[33] M. Ntampaka, C. Avestruz, S. Boada, J. Caldeira, J. Cisewski-Kehe, R. Di Stefano, C. Dvorkin, A. E. Evrard, A. Farahi, D. Finkbeiner, et al., *The Role of Machine Learning in the Next Decade of Cosmology*, arXiv preprint, https://arxiv.org/abs/1902.10159, 2019.

[34] A. Odena, J. Buckman, C. Olsson, T. B. Brown, C. Olah, C. Raffel, and I. Goodfellow, *Is Generator Conditioning Causally Related to GAN Performance?*, arXiv preprint, https://arxiv.org/abs/1802.08768, 2018.

[35] N. Park, A. Anand, J. R. A. Moniz, K. Lee, T. Chakraborty, J. Choo, H. Park, and Y. Kim, *MMGAN: Manifold Matching Generative Adversarial Network for Generating Images*, arXiv preprint, https://arxiv.org/abs/1707.08273, 2017.

[36] B. Poole, A. A. Alemi, J. Sohl-Dickstein, and A. Angelova, *Improved Generator Objectives for GANs*, arXiv preprint, https://arxiv.org/abs/1612.02780, 2016.

[37] A. Radford, L. Metz, and S. Chintala, *Unsupervised representation learning with deep convolutional generative adversarial networks*, in Proceedings of the International Conference on Learning Representations (ICLR), 2016.

[38] K. Roth, A. Lucchi, S. Nowozin, and T. Hofmann, *Stabilizing training of generative adversarial networks through regularization*, in Advances in Neural Information Processing Systems, 2017, pp. 2018–2028.

[39] W. Ruan, X. Huang, and M. Kwiatkowska, *Reachability Analysis of Deep Neural Networks with Provable Guarantees*, arXiv preprint, https://arxiv.org/abs/1805.02242, 2018.

[40] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, *Improved techniques for training GANs*, in Advances in Neural Information Processing Systems, 2016, pp. 2234–2242.

[41] M. J. Smith and J. E. Geach, *Generative deep fields: Arbitrarily sized, random synthetic astronomical images through deep learning*, Monthly Not. Roy. Astronom. Soc., 490 (2019), pp. 4985–4990.

[42] Z. Yang, X. Li, L. Catherine Brinson, A. N. Choudhary, W. Chen, and A. Agrawal, *Microstructural materials design via deep adversarial learning methodology*, J. Mech. Des., 140 (2018).

[43] X. Yi, E. Walia, and P. Babyn, *Generative adversarial network in medical imaging: A review*, Med. Image Anal., 58 (2019), 101552.

[44] J. Zhang, B. Kailkhura, and T. Han, *Leveraging Uncertainty from Deep Learning for Trustworthy Materials Discovery Workflows*, arXiv preprint, https://arxiv.org/abs/2012.01478, 2020.