A GEOMETRIC PERSPECTIVE ON RECURSIVE SYNTHETIC TRAINING

Patrick Batsell
Department of Computational and Applied Math and Operations Research
Rice University
Houston, TX, USA
pb52@rice.edu

Thomas Walker & Richard Baraniuk
Department of Electrical and Computer Engineering
Rice University
Houston, TX, USA
{thomas.walker,richb}@rice.edu

ABSTRACT

Scaling high-quality datasets to improve generative model quality is effective, but is becoming increasingly challenging due to data scarcity and contamination. Trying to alleviate this by naively bootstrapping generative models by training on synthetic data results in significant quality degradation and a collapse in sample diversity. In this paper, we study the negative effects of synthetic data on the geometry of deep generative networks (DGNs) to understand how to go beyond naive synthetic data utilization. Through empirical simulations, we show that retraining on synthetic data leads to DGNs with low-quality singular vectors and input-output Jacobians with low effective rank. Using these insights, we develop a strategy to generate synthetic data from a DGN to improve its quality through negative guidance.

1 INTRODUCTION

The recent trajectory of progress in generative modeling has been defined by a "scaling first" paradigm. From large language models to high-fidelity diffusion frameworks, the leap in performance is largely attributed to the massive expansion of parameters and computational budgets (Kaplan et al., 2020; Henighan et al., 2020). At the heart of this scaling success lies the requirement of data. As models continue to grow in capacity, the availability of high-quality training data is rapidly becoming a bottleneck for future advancement.

To navigate this scarcity, using synthetically generated data appears as an attractive, low-cost solution for continued training. Yet, this approach introduces a parasitic feedback loop. Recent studies have shown that when generative models are trained on their own outputs, they are plagued by Model Autophagy Disorder (MADness) (Alemohammad et al., 2024) and "model collapse" (Shumailov et al., 2024). Consequently, synthetic data cannot be treated as a direct substitute for realistic data; rather, it must be handled with distinct strategies to improve, rather than degrade, model performance.

In this paper, we adopt a geometric perspective on deep generative networks (DGNs) to understand the causes and effects of MADness. Unlike observing global distribution metrics (e.g., FID), we analyze the local geometry of the latent space via the input-output Jacobians of DGNs. We determine that recursive training on synthetic data leads to a collapse in the effective ranks of these Jacobians and the proliferation of low-quality artifacts in their left singular vectors (as seen in Figure 1). This understanding offers mechanisms to identify the early signatures of MADness and improved strategies for leveraging synthetic data to improve model performance.

2 BACKGROUND

Continuous Piecewise Affine Mappings. A mapping $g : \mathbb{R}^h \to \mathbb{R}^d$ is continuous piecewise affine (CPA) if it is continuous and can be written in the form $g(\boldsymbol{z}) = \sum_{\omega \in \Omega} (\boldsymbol{A}_\omega \boldsymbol{z} + \boldsymbol{b}_\omega) \mathbb{I}_{\{\boldsymbol{z} \in \omega\}}$, where Ω is a partition of the input domain $\mathbb{R}^h$ into convex polytopes, $\boldsymbol{A}_\omega \in \mathbb{R}^{d \times h}$, and $\boldsymbol{b}_\omega \in \mathbb{R}^d$. That is, in a local region of the input space $\omega \subseteq \mathbb{R}^h$, the mapping g corresponds to a specific affine transformation, as seen in Figure 1. For a given point $\boldsymbol{z} \in \mathbb{R}^h$, we will denote by $\omega(\boldsymbol{z}) \in \Omega$ the linear region of g that contains $\boldsymbol{z}$.

Deep Generative Networks. A deep generative network (DGN) is a parametrized mapping g_θ from a latent space $\mathbb{R}^h$ to an observation space $\mathbb{R}^d$; constructed as a composition of affine transformations followed by nonlinearities. If the nonlinearities are CPA (e.g., the ReLU activation function), then the DGN is a CPA mapping (Balestriero & Baraniuk, 2018). To develop our ideas and intuition, we will henceforth consider CPA DGNs, however, our results extend to general DGNs since they involve the input-output Jacobians of these mapping which exist for any differentiable DGN.

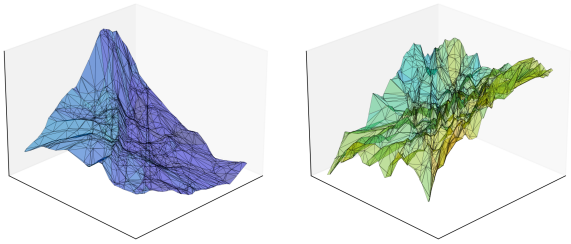


Figure 1: The DGN of a VAE generative model exhibits MADness geometrically as an increasingly *jagged* mapping. Here we train a VAE on a two-dimensional circular distribution in a synthetic loop. For the zeroth and second generations, we visualise the continuous piecewise affine (CPA) mapping of the DGN from the latent space to the first output dimension. The colours of the hyperplanes are given by the Frobenius norm of the $\boldsymbol{A}_\omega$ matrix. The scale of this colouring is the same across each plot. For more experimental details refer to Section A.1.

Output Manifold of DGNs. DGNs are trained to map a distribution on the latent space $\mathbb{R}^h$ to a distribution on the observation space $\mathbb{R}^d$. The distribution on the latent space has a simple form (i.e., Gaussian or Uniform), and the distribution on the observation space is matched to the distribution of some data.

The local linearity of a DGN means that it is possible to analytically characterize the nature of the learned distribution on the observation space given a known distribution on the latent space. Let p_{lat} refer to the distribution on the latent space, and p_{rec} be the distribution learned by the DGN g_θ. Suppose the mild constraints that $\boldsymbol{A}_\omega$ is full rank for every $\omega \in \Omega$ and $g(\omega) \cap g(\omega') = \emptyset$ for any distinct $\omega, \omega' \in \Omega$ (i.e., the map has no self-intersections).

Theorem 1 (Humayun et al. 2022)**.** *We have*

$$p_{\text{rec}}(\boldsymbol{z}) = \sum_{\omega \in \Omega} \frac{1}{\sqrt{\det\left(\boldsymbol{A}_\omega^\top \boldsymbol{A}_\omega\right)}} p_{\text{lat}}\left(\left(\boldsymbol{A}_\omega^\top \boldsymbol{A}_\omega\right)^{-1} \boldsymbol{A}_\omega^\top \left(\boldsymbol{z} - \boldsymbol{b}_\omega\right)\right) \mathbb{I}_{\{\boldsymbol{z} \in \omega\}}.$$

From Theorem 1, we observe a direct connection between the learned distribution of a DGN, and the singular values of its input-output Jacobians.

Model Autophagy Disorder. DGNs are congruently trained with encoders that map training data into the latent space, in a manner that conforms to the desired distribution p_{lat}. Arriving at a performant DGN, requires sufficient training data for the encoder to appropriately map samples into the latent space which can then be used by the DGN to learn the appropriate reconstruction mapping g_θ. Naively bootstrapping this process by feeding in synthetic samples from previous constructions of g_θ into the training data to facilitate the construction of an improved g_θ has been thoroughly dismissed with the identification of the Model Autophagy Disorder (MADness) (Alemohammad et al., 2024).

MADness describes how the process of iteratively training a generative model on its synthetic generations leads to a *degradation* of its performance on both a local and global scale. Globally, it results

in mode collapse (Liu et al., 2019) which describes p_{rec} becoming an increasingly concentrated distribution. Locally, it results in low-quality artefacts in the generated samples.

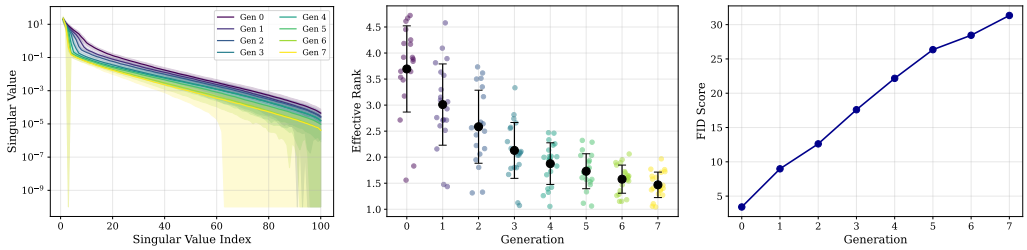


Figure 2: The effective ranks of a DGN trained on MNIST diminish during MADness. Here, statistics Jacobians of a DGN, forming part of a GAN, are visulized as the GAN undergoes a synthetic training loop on the MNIST dataset. On the left, we visualize the average singular values, along with their standard deviations, at different generations. In the centre plot, we visualize the effective ranks at different generations. In the right, we record the FID score of the samples generated by the DGN at different generations. For more experimental details, refer to Section A.2.

3 The Geometry of MADness

In this section, we study the geometric realization of MADness by taking inspiration from Theorem 1, which emphasises the relationship between the output manifold of a DGN and the singular value spectrum of its Jacobians. Recall that $\boldsymbol{A}_\omega$ is just the input-output Jacobian of g_θ computed at a point in ω.

Our focus will be on the geometry of the DGN as it is retrained on synthetically generated samples from previous generations. Generation 0 corresponds to a DGN $g_{\theta^{(0)}}$ trained on real data. Generation n corresponds to a DGN $g_{\theta^{(n)}}$ trained on synthetic samples generated from $g_{\theta^{(n-1)}}$.

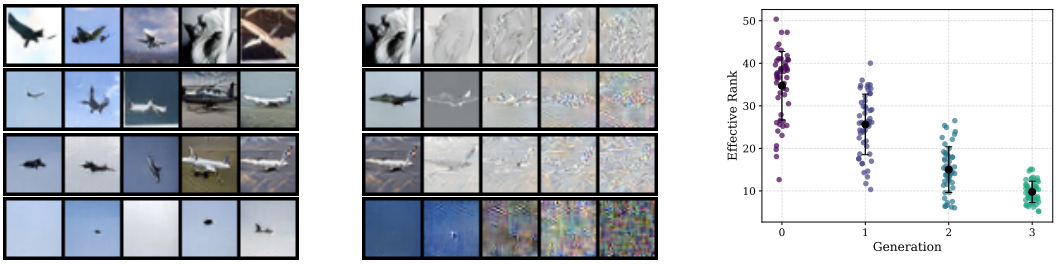
As a preliminary example, we consider the DGN of a Variational Autoencoder (Kingma & Welling, 2022) trained on a circular distribution in two-dimensional space. In Figure 1, we visualize the CPA mapping of the learned DGN of the zeroth and second generation, at which point the VAE demonstrates mode collapse. Qualitatively, the mapping of the DGN exhibits a large Lipschitz constant as it suffers from MADness. In the context of Theorem 1, this results in an output distribution p_{rec} which is degenerate in local regions.

In higher dimensions, we study the singular value decompositions of the matrices $\boldsymbol{A}_\omega$ of vectors sampled from the latent space. We look at the left singular vectors $\boldsymbol{u}_i \in \mathbb{R}^d$ – where i denotes the index of the singular vector – and the effective rank as given by $\operatorname{er}(\boldsymbol{A}_\omega) = \exp\left(-\sum_{i=1}^{d} s_i \log(s_i)\right)$ where $s_i = \frac{\sigma_i}{\sum_{j=1}^{d} \sigma_j}$ for σ_i the i^{th} singular value of $\boldsymbol{A}_\omega$ (Roy & Vetterli, 2007).

In Figure 2 we corroborate the observations from Figure 1 for the DGN of a Generative Adversarial Network (GAN) (Goodfellow et al., 2014) reconstructing the MNIST data distribution. As the DGN exhibits MADness – exhibited by the deteriorating FID score (Heusel et al., 2017) in the right plot of Figure 2 – the effective ranks of the Jacobians of the DGN diminish.

Next, we study the left singular vectors for the DGN obtained from a BigGAN (Brock et al., 2019) trained on CIFAR10 (Krizhevsky & Hinton, 2009). From Figure 3 it becomes clearer how the symptoms of MADness (i.e., quality degradation and mode collapse) manifest.

At generation 0, the DGN already exhibits a preference to generating "simpler" images, and the high-index singular vectors exhibit some noise. The preference for "simpler" images is virtue of the phenomenon of spectral collapse which is known to hinder the training of these models (Liu et al., 2019), and describes how the diminishing of high-order spectral values (i.e., diminishing effective rank) leads to mode collapse. This is a result of the known tendency of deep architectures for learning low-rank solutions (Feng et al., 2022).



Synthetic Samples | Samples w/ Singular Vectors | Effective Rank

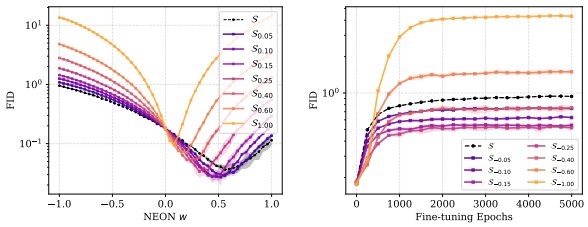
Figure 3: The singular vectors of a DGN become increasing noisy throughout the synthetic training loop. Here we train a BigGAN in a synthetic loop on the CIFAR10 dataset. In the left plot we visualise synthetic samples ordered from low to high effective ranks of their corresponding Jacobians. In the centre plot we visualise synthetic samples along with four singular vectors of their corresponding Jacobians for equally spaced orders. In the right plot we visualise the decreasing effective rank across the generations. For more experimental details, refer to Section A.3.

Throughout the synthetic training loop, the effective ranks of the Jacobians diminish and noise becomes present in increasingly lower-order singular vectors. The symptom of this is that the model experiences mode collapse, with the quality of samples reducing as noisy artifacts become visible.

4 Experiments

NEON (Alemohammad et al., 2026) is a known strategy to capitalize on the degeneracy of synthetic data to improve the performance of DGN. It operates by fine-tuning g_θ on synthetic data $\mathcal{S}$–yielding g_{θ_s}–to identify a direction in parameter space $\theta - \theta_s$ for updating the DGN's parameters $\theta_{\text{Neon}} = \theta + w\,(\theta - \theta_s)$, where $w > 0$. Although simple, NEON has led to noticeable improvements in state-of-the-art DGNs. Our investigations into the geometric properties of MADness can augment $\mathcal{S}$ to yield parameter updates that more effectively improve the DGN's performance.

Our approach for generating synthetic data is as follows. Set $\alpha \in [0, 1]$. Sample $\boldsymbol{z} \in \mathbb{R}^h$. Compute $\boldsymbol{A}_{\omega(\boldsymbol{z})}$ and $\boldsymbol{b}_{\omega(\boldsymbol{z})}$. Compute the singular value decomposition of $\boldsymbol{A}_{\omega(\boldsymbol{z})}$, namely $\boldsymbol{A}_{\omega(\boldsymbol{z})} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, where $\boldsymbol{U} \in \mathbb{R}^{d\times r}$, $\boldsymbol{\Sigma} = \operatorname{diag}(\sigma_1, \ldots, \sigma_r) \in \mathbb{R}^{r\times r}$, $\boldsymbol{V} \in \mathbb{R}^{r\times h}$. "Collapse" the distribution $\{\sigma_1, \ldots, \sigma_r\}$ using α as described in Section B, to yield $\{\tilde{\sigma}_1, \ldots, \tilde{\sigma}_r\}$. Compute $\tilde{\boldsymbol{A}}_{\omega(\boldsymbol{z})} = \boldsymbol{U}\widetilde{\boldsymbol{\Sigma}}\boldsymbol{V}^\top$ where $\widetilde{\boldsymbol{\Sigma}} = \operatorname{diag}(\tilde{\sigma}_1, \ldots, \tilde{\sigma}_r)$. Let $\boldsymbol{s} = \tilde{\boldsymbol{A}}_{\omega(\boldsymbol{z})}\boldsymbol{z} + \boldsymbol{b}_{\omega(\boldsymbol{z})}$ and add to $\mathcal{S}_\alpha$. Repeat to build a synthetic dataset $\mathcal{S}_\alpha$.



(a) (b)

Figure 4: The geometrical insights of Section 3 lead to more effective applications of NEON and fine-tuning. In Figure 4a, we implement NEON with different parameters w on a VAE training on a five-dimensional Gaussian distribution. In Figure 4b, we fine-tune this VAE on synthetically generated data. For more experimental details refer to Section A.6.

The collapsing of the singular value distribution of $\boldsymbol{A}_{\omega(\boldsymbol{z})}$ allows for the DGN to correct itself under the negative guidance provided by NEON. In Figure 4a, we observe that with $\alpha \approx 0.1$, NEON yields a DGN with a lower FID as compared to using standard synthetic data.

Equally, the spectrum $\{\sigma_1, \ldots, \sigma_r\}$ can be "flattened" to dampen the severity of MADness. In Section B, we outline this procedure which utilizes a parameter $\beta \in [0, 1]$. For illustrative purposes, we set $\alpha = -\beta$, such that this can be compared to synthetic samples obtained through "collapsing" the spectrum by considering the sets $\mathcal{S}_\alpha$ for $\alpha \in [-1, 1]$. In Figure 4b we consider the effects of

fine-tuning a VAE trained on synthetic data $\mathcal{S}_\alpha$. We observe that relative to standardly generated synthetic data $\mathcal{S}$, synthetic data $\mathcal{S}_\alpha$ for $\alpha \in [-0.4, 0]$ dampens the severity of MADness.

5 Discussion

Training on synthetic data has the effect of increasing its Lipschitz constant of DGNs. This distorts its output distribution, amplifies errors in the singular vectors of its Jacobians, and leads to mode collapse. Using these insights, we can intervene on how synthetic samples are generated from a DGN to improve performance through negative guidance.

Acknowledgments

This work was supported by ONR grant N00014-23-1-2714, DOE grant DE-SC0020345, DOI grant 140D0423C0076, and a Google Cloud Computing Award.

REFERENCES

Sina Alemohammad, Josue Casco-Rodriguez, Lorenzo Luzi, Ahmed Imtiaz Humayun, Hossein Babaei, Daniel LeJeune, Ali Siahkoohi, and Richard Baraniuk. Self-consuming generative models go MAD. In *International Conference on Learning Representations*, 2024.

Sina Alemohammad, Zhangyang Wang, and Richard Baraniuk. Neon: Negative Extrapolation From Self-training Improves Image Generation. In *The Fourteenth International Conference on Learning Representations*, 2026.

Randall Balestriero and Richard Baraniuk. A Spline Theory of Deep Learning. In *Proceedings of the 35th International Conference on Machine Learning*. PMLR, July 2018.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019.

Ruili Feng, Kecheng Zheng, Yukun Huang, Deli Zhao, Michael Jordan, and Zheng-Jun Zha. Rank Diminishing in Deep Neural Networks. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, 2022.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks, 2014. arXiv: 1406.2661.

Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B. Brown, Prafulla Dhariwal, Scott Gray, Chris Hallacy, Benjamin Mann, Alec Radford, Aditya Ramesh, Nick Ryder, Daniel M. Ziegler, John Schulman, Dario Amodei, and Sam McCandlish. Scaling Laws for Autoregressive Generative Modeling, 2020. arXiv:2010.14701.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, 2017.

Ahmed Imtiaz Humayun, Randall Balestriero, and Richard Baraniuk. MaGNET: Uniform Sampling from Deep Generative Network Manifolds Without Retraining. In *International Conference on Learning Representations*, January 2022.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling Laws for Neural Language Models, January 2020. arXiv:2001.08361.

Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes, 2022. arXiv:1312.6114.

Alex Krizhevsky and Geoffrey Hinton. Learning Multiple Layers of Features from Tiny Images. Technical report, University of Toronto, Toronto, Ontario, 2009.

Kanglin Liu, Guoping Qiu, Wenming Tang, and Fei Zhou. Spectral Regularization for Combating Mode Collapse in GANs. In *IEEE/CVF International Conference on Computer Vision*, pp. 6381–6389, 2019.

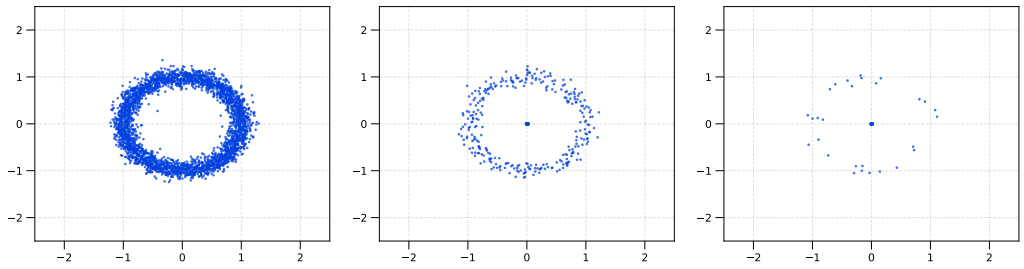
Olivier Roy and Martin Vetterli. The Effective Rank: A Measure of Effective Dimensionality. In *15th European Signal Processing Conference*, pp. 606–610, 2007.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. AI Models Collapse when Trained on Recursively Generated Data. *Nature*, 631(8022):755–759, July 2024.

A Experimental Details

A.1 Two-Dimensional Gaussian Partial Synthetic Data Loop

Dataset. The ground-truth data consists of 2000 samples equally spaced around the circumference of a two-dimensional circle with radius one, centered at the origin, with Gaussian noise applied in the radial component. A visualisation of the data distribution used to train each generation is shown in Figure 5. At each subsequent generation, 1800 synthetic samples are generated, and 200 are retrained from the initial ground-truth distribution.



Generation 0 Generation 1 Generation 2

Figure 5: A visualization of the training data used to train each generation of the VAE of Figure 1. The training data at generation n, for $n \geq 1$, contains 200 samples from the training data of generation 0 and 1800 synthetic samples generated from the VAE of generation $n-1$.

Architecture. We use a Variational AutoEncoder (VAE).

- Encoder: A 3-layer Leaky-ReLU MLP with constant width 64 mapping a two-dimensional input space to a two-dimensional latent space.
- Decoder: A 3-layer Leaky-ReLU MLP with constant width 64 mapping a two-dimensional latent space to a two-dimensional data space.

Training. The VAE is trained using the Adam optimizer with a learning rate of 0.001. The loss function is constructed from a reconstruction term (i.e., squared-error between the input and output of the VAE) and a KL-divergence term (i.e., the KL divergence between the latent space and the two-dimensional isotropic Gaussian distribution). The model is trained for 3000 iterations.

A.2 MNIST GAN Full Synthetic Loop

Dataset. MNIST consists of 60,000 training images of 28×28 grayscale hand-written digits across 10 classes (i.e., the ten different digits).

Architecture. We use a conditional GAN with class-label embedding.

- Generator: A 4-layer Leaky-ReLU MLP that takes a 100-dimensional latent vector concatenated with a 10-dimensional class embedding. The layer sizes are $[110, 128, 256, 512, 1024, 784]$. Batch normalization is applied between each hidden layer, and the $\tanh$ activation function is applied after the final layer.
- Discriminator: A 4-layer Leaky-ReLU MLP that takes a flattened image concatenated with a 10-dimensional class embedding. The layer sizes are $[794, 512, 512, 512, 512, 1]$. Dropout is applied between each hidden layer, and the $\mathrm{sigmoid}$ activation function is applied after the final layer.

Training. We train the GAN using the binary cross-entropy loss. The optimizer is Adam with learning rate 2×10^{-4} and momentum parameters $\beta_1 = 0.5$, $\beta_2 = 0.999$. Each generation is trained for 50 epochs with a batch size 128. Images are normalized to $[-1, 1]$.

Synthetic Loop. We run the fully synthetic loop for 8 generations. At each generation $n \geq 1$, we generate 60,000 synthetic samples from the generator $g_{\theta^{(n-1)}}$ trained in the previous generation. The model $g_{\theta^{(n)}}$ is then trained exclusively on these synthetic samples.

Jacobian Analysis. For each generation, we compute the Jacobian of the generator with respect to the 100-dimensional latent input for 100 randomly sampled latent vectors using PyTorch's automatic differentiation (`torch.autograd.functional.jacobian`). We perform singular value decomposition (SVD) on each Jacobian matrix and compute the effective rank.

FID Computation. We compute the Frechet Inception Distance (FID) using a LeNet classifier trained on MNIST as the feature extractor. Features are extracted from the 84-dimensional penultimate layer. The FID is computed as:

$$\text{FID} = \|\mu_r - \mu_g\|^2 + \text{tr}\left(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}\right)$$

where (μ_r, Σ_r) and (μ_g, Σ_g) are the mean and covariance of the real and generated feature distributions, respectively.

A.3 BIGGAN CIFAR10

The training of the base model matches the default settings of the following GitHub repository: `https://github.com/ajbrock/BigGAN-PyTorch`. For subsequent generations in the synthetic loop, we sample 50,000 images from the previous generation generative model and train a random initialization architecture in the same ways as the base model.

A.4 FASHIONMNIST GAN FULL SYNTHETIC LOOP

We extend our MNIST experiments to the FashionMNIST dataset to demonstrate the generality of our findings across different data distributions. The results can be observed in Figure 6.

Dataset. FashionMNIST consists of 60,000 training images of 28×28 grayscale clothing items across 10 classes (t-shirt, trouser, pullover, dress, coat, sandal, shirt, sneaker, bag, ankle boot).

Architecture. We use the same conditional GAN architecture as that of Section A.2.

Training. We use the same training procedure as that of Section A.2.

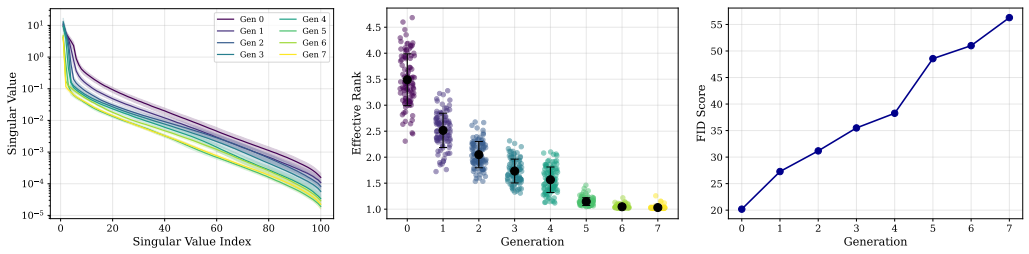


Figure 6: The effective ranks of a DGN trained on FashionMNIST diminish during MADness, demonstrating the generality of our findings across different data distributions. Here the Jacobians of a DGN, forming part of a GAN, are visualized as the GAN undergoes a fully synthetic training loop on the FashionMNIST dataset. On the left, we visualize the average singular values, along with their standard deviations, of the Jacobians of the DGN at different generations. In the centre plot, we visualize the effective ranks of the Jacobians of the DGN at different generations. On the right, we record the FID score of the samples generated by the DGN at different generations.

A.5 MNIST GAN SYNTHETIC AUGMENTATION LOOP EXPERIMENTS

We investigate the synthetic augmentation loop, where models are trained on a combination of fixed real data and synthetic data from previous generations. The results can be observed in Figure 7.

Architecture. We use the same conditional GAN architecture as that of Section A.2.

Training. We use the same training procedure as that of Section A.2.

Synthetic Augmentation Loop. Unlike the fully synthetic loop, the real dataset remains fixed and present at every generation, while synthetic data is replaced with new samples from the most recent generation.

- Generation 0: Train on real data only (60k samples)
- Generation 1: Train on real 25% real data + 75% synthetic from $g_{\theta^{(0)}}$.
- Generation 2: Train on real 25% real data + 75% synthetic from $g_{\theta^{(1)}}$.
- Generation n: Train on real 25% real data + 75% synthetic from $g_{\theta^{(n-1)}}$.

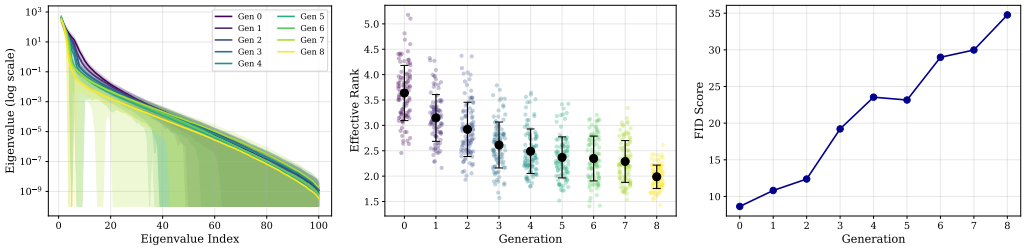


Figure 7: The effective ranks of a DGN diminish during MADness even when 25% of the training data remains real. Here the Jacobians of a DGN, forming part of a GAN, are visualized as the GAN undergoes a synthetic augmentation loop on MNIST with a constant 75% synthetic to 25% real data ratio. On the left, we visualize the average singular values, along with their standard deviations, of the Jacobians of the DGN at different generations. In the centre plot, we visualize the effective ranks of the Jacobians of the DGN at different generations. On the right, we record the FID score of the samples generated by the DGN at different generations. The presence of real data slows but does not prevent degradation, with effective rank declining from 3.64 to 1.99 across 9 generations.

A.6 SELF-IMPROVING DGN WITH NEON

Dataset. The ground-truth data is samples of 5000 points from a five-dimensional isotropic Gaussian. For synthetic fine-tuning, 2000 synthetic samples are generated from the DGN.

Architecture. The generative model is a VAE.

- Encoder: A 2-layer ReLU MLP with layer sizes $[5, 128]$.
- Latent Space: The output of the encoder is then mapped to distributional parameters of dimension five which are used to generate a latent vector by transforming a sample from a standard isotropic Gaussian in five dimensions.
- Decoder: A 3-layer ReLU MLP with layer sizes $[5, 128, 5]$ which maps a latent vector back to the data space.

Training.

- Base Model: Trained across 10,000 epochs, at a learning rate of 1×10^{-4} using the Adam optimizer. The loss function is a linear combination of a reconstruction loss and a KL-divergence to a standard isotropic Gaussian on the latent space.

- Synthetic Fine-tuning: The trained base model is fine-tuned on a set of synthetic data for 5000 epochs using the Adam optimizer at a learning rate of 1×10^{-5}.

Weight Merging. For the base model with parameters θ, after fine-tuning to obtain parameters θ_s we then update the parameters of the base model to $\theta_{\text{NEON}} = \theta + w(\theta - \theta_s)$. In our experiments, we consider w taking 50 equally spaced values in the range $[-1, 1]$.

Synthetic Data Generation. In our experiments, we fix the seed used to train the base model and then consider fine-tuning it on synthetic data drawn using $\alpha, \beta \in \{0.0, 0.05, 0.1, 0.15, 0.25, 0.4, 0.6, 1.0\}$. For values of α and β we draw three samples using different random seeds, but keeping the base model the same.

Further Analysis. With Figure 8 we show the singular value spectra and the effective ranks of the DGN that achieved the lowest FID value for each value of α in Figure 4a. We can see that higher effective ranks correlate with lower FID values in Figure 4a.

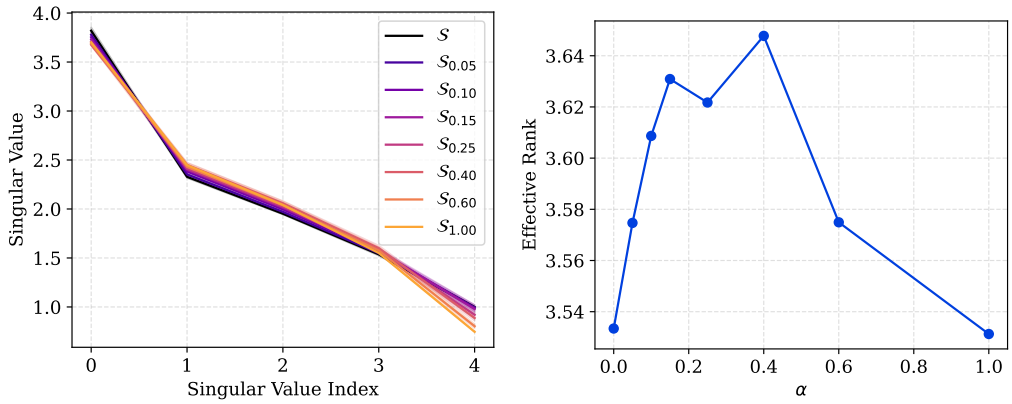


Figure 8: For each value of α in Figure 4a, we take the weights of the DGN that achieves the lowest FID and compute the spectra of its Jacobians and the effective rank of its Jacobians on a set of randomly sampled latent vectors.

B Synthetic Data Generation

Collapsing the Spectrum. As mentioned in Section 4, our proposed procedure for generating a set of synthetic data $\mathcal{S}_\alpha$ on which to fine-tune the DGN to find the direction in which to apply negative guidance, requires "collapsing" the singular value spectrum of the Jacobians of the DGN using the parameter α. Namely, given the singular value spectrum of $\boldsymbol{A}_{\omega(\boldsymbol{z})}$ as $\{\sigma_1, \ldots, \sigma_r\}$, we obtain the collapsed spectrum $\{\tilde{\sigma}_1, \ldots, \tilde{\sigma}_r\}$ as

$$\tilde{\sigma}_i = \begin{cases} \sqrt{(1-\alpha)\sigma_1^2 + \alpha E} & i = 1 \\ \sqrt{1-\alpha}\sigma_i & i \geq 2, \end{cases}$$

where $E = \sum_{i=1}^{r} \sigma_i^2$. Thus, when $\alpha = 0$ no change is made to the singular value spectrum, but when $\alpha = 1$ all the energy of the spectrum is concentrated on the first singular value.

The consequences this operation has on the spectra and singular values of the Jacobians of the DGN, as well as the subsequent generated sample for the setting of Section A.6 are shown in Figure 9. Interestingly, we observe, that in this setting, increasing α has a mode-seeking effect. Namely, as $\alpha \to 1$ the distribution of the synthetic samples seems to collapse to a narrow distribution. Thus, the observed improvement this operation has on the effectiveness of NEON, see Figure 4a, is not surprising as it is precisely mode-seeking synthetic data for which NEON was theoretically demonstrated to work (Alemohammad et al., 2026).

Flattening the Spectrum. A similar procedure can be done to "flatten" the singular value spectrum of the Jacobians of the DGN using a parameter $\beta \in [0, 1]$. Given the singular value spectrum of $\boldsymbol{A}_{\omega(\boldsymbol{z})}$ as $\{\sigma_1, \ldots, \sigma_r\}$, we obtain the flattened spectrum $\{\tilde{\sigma}_1, \ldots, \tilde{\sigma}_r\}$ as

$$\tilde{\sigma}_i = (1 - \beta)\sigma_i + \frac{\beta}{r}E.$$

Thus, when $\beta = 0$ there is no change, but when $\beta = 1$ the energy of the spectrum is equally distributed across the singular values.

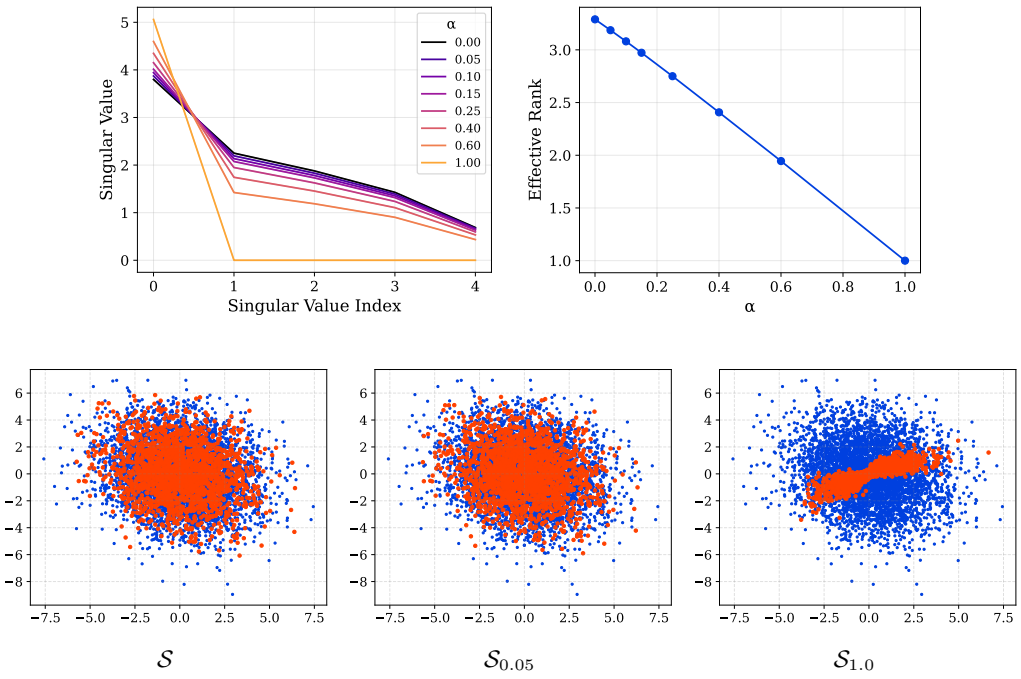


Figure 9: In the top row, we demonstrate the effect of the parameter α has on the spectra of Jacobians of a DGN. In the bottom row, we visualize the nature of the synthetic samples $\mathcal{S}_\alpha$ with respect to α by plotting the first two-dimensions of the ground-truth data (blue) and the first two-dimensions of the synthetic data (orange).