Is Generator Conditioning Causally Related to GAN Performance?

Augustus Odena [1]   **Jacob Buckman** [1]   **Catherine Olsson** [1]   **Tom B. Brown** [1]   **Christopher Olah** [1]   **Colin Raffel** [1]
Ian Goodfellow [1]

Abstract

Recent work (Pennington et al., 2017) suggests that controlling the entire distribution of Jacobian singular values is an important design consideration in deep learning. Motivated by this, we study the distribution of singular values of the Jacobian of the generator in Generative Adversarial Networks (GANs). We find that this Jacobian generally becomes ill-conditioned at the beginning of training and that the average (with $z \sim p(z)$) conditioning of the generator is highly predictive of two other ad-hoc metrics for measuring the "quality" of trained GANs: the Inception Score and the Frechet Inception Distance (FID). We test the hypothesis that this relationship is causal by proposing a "regularization" technique (called Jacobian Clamping) that softly penalizes the condition number of the generator Jacobian. Jacobian Clamping improves the mean Inception Score and the mean FID for GANs trained on several datasets and greatly reduces inter-run variance of the aforementioned scores, addressing (at least partially) one of the main criticisms of GANs.

1. Introduction

Generative Adversarial Networks (or GANs) are a promising techique for building flexible generative models (Goodfellow et al., 2014). There have been many successful efforts to scale them up to large datasets and new applications (Denton et al., 2015; Radford et al., 2015; Odena et al., 2016; Zhang et al., 2016; Karras et al., 2017; Miyato & Koyama, 2018). There have also been many efforts to better understand their training procedure, and in particular to understand various pathologies that seem to plague that training procedure (Metz et al., 2016; Arora et al., 2017; Heusel et al., 2017; Nagarajan & Kolter, 2017; Arjovsky

& Bottou, 2017). The most notable of these pathologies — "mode collapse" — is characterized by a tendency of the generator to output samples from a small subset of the modes of the data distribution. In extreme cases, the generator will output only a few unique samples or even just the same sample repeatedly. Instead of studying this pathology and others from a probabilistic perspective, we study the distribution of the squared singular values of the input-output Jacobian of the generator. Studying this quantity allows us to characterize GANs in a new way — we find that it is predictive of other GAN performance measures. Moreover, we find that by controlling this quantity, we can improve average-case performance measures while greatly reducing inter-run variance of those measures. More specifically, this work makes the following contributions:

- We study the squared singular values of the generator Jacobian at individual points in the latent space. We find that the Jacobian generally becomes ill-conditioned quickly at the beginning of training, after which it tends to fall into one of two clusters: a "good cluster" in which the condition number stays the same or even gradually decreases, and a "bad cluster", in which the condition number continues to grow.

- We discover a strong correspondence between the conditioning of the Jacobian and two other quantitative metrics for evaluating GAN quality: the Inception Score and the Frechet Inception Distance. GANs with better conditioning tend to perform better according to these metrics.

- We provide evidence that the above correspondence is causal by proposing and testing a new regularization technique, which we call Jacobian Clamping. We show that you can constrain the conditioning of the Jacobian relatively cheaply[1] and that doing so improves the mean values and reduces inter-run variance of the values for the Inception Score and FID.

2. Background and Notation

Generative Adversarial Networks: A generative adversarial network (GAN) consists of two neural networks

---

[1]Google Brain. Correspondence to: Augustus Odena <augustusodena@google.com>.

---

[1]The Jacobian Clamping algorithm doubles the batch size.

Normally, GAN training is stochastic: sometimes GANs train well and sometimes they fail.

With **Jacobian Clamping**, GANs consistently have lower condition numbers and perform well.
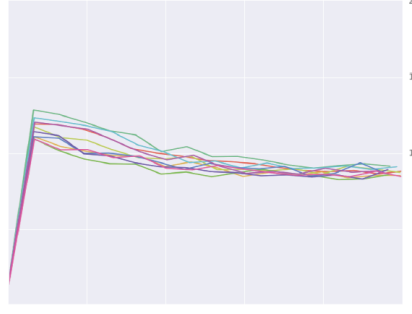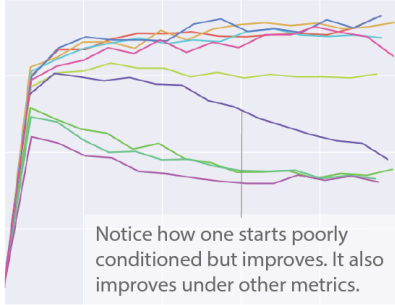
The mean Log **Condition Number** of the metric tensor describes how consistently distance in the latent space maps to distance in the output space.

Notice how one starts poorly conditioned but improves. It also improves under other metrics.
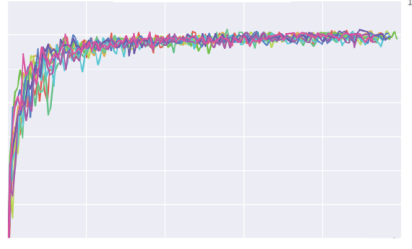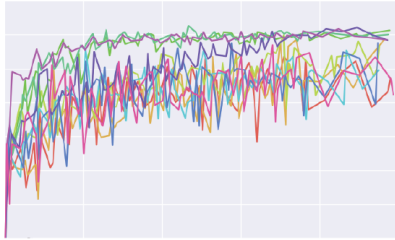
Classifier Score is a standard metric of GAN performance. A high score means the GAN is producing samples that are confidently classified as each class.
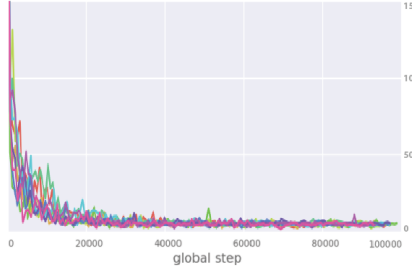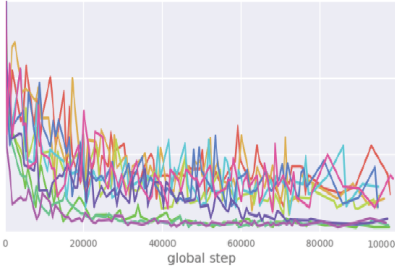
Frechet Distance is another standard metric of GAN performance. Lower is better.

Figure 1. MNIST Experiments. Left and right columns correspond to 10 runs without and with Jacobian Clamping, respectively. Within each column, each run has a unique color. Top to bottom, rows correspond to mean log-condition number, Classifier Score, and Frechet Distance. Note the dark purple run in the left column: the generator moves from the ill-conditioned cluster to the well-conditioned cluster while also moving from the low-scoring cluster to the high-scoring cluster.

trained in opposition to one another. The generator G takes as input a random noise vector $z \sim p(z)$ and outputs a sample $G(z)$. The discriminator D receives as input either a training sample or a synthesized sample from the generator and outputs a probability distribution $D(x)$ over possible sample sources. The discriminator is then trained to maximize the following cost:

$$L_D = - \mathop{\mathbb{E}}_{x \sim p_{\text{data}}} [\log D(x)] - \mathop{\mathbb{E}}_{z \sim p(z)} [\log(1 - D(G(z)))] \quad (1)$$

while the generator is trained to minimize[2]:

$$L_G = - \mathop{\mathbb{E}}_{z \sim p(z)} [\log D(G(z))] \quad (2)$$

_____

[2] This formulation is known as a "Non-Saturating GAN" and is the formulation in wide use, but there are others. See Goodfellow et al. (2014) for more details.

Inception Score and Frechet Inception Distance: In this work we will refer extensively to two[3] "scores" that have been proposed to evaluate the quality of trained GANs. Both make use of a pre-trained image classifier. The first is the Inception Score (Salimans et al., 2016), which is given by:

$$\exp \left(\mathbb{E}_{\mathbf{x} \in P_\theta} [KL(p(y|\mathbf{x}) \| p(y))] \right) \quad (3)$$

where $\mathbf{x}$ is a GAN sample, $p(y|\mathbf{x})$ is the probability for labels y given by a pretrained classifier on $\mathbf{x}$, and $p(y)$ is the overall distribution of labels in the generated samples

_____

[3] We elect not to use the technique described in Wu et al. (2016) for reasons explained in Grover et al. (2017).

(according to that classifier).

The second is the Frechet Inception Distance (Heusel et al., 2017). To compute this distance, one assumes that the activations in the coding layer of the pre-trained classifier come from a multivariate Gaussian. If the activations on the real data are $N(m, C)$ and the activations on the fake data are $N(m_w, C_w)$, the FID is given by:

$$\|m - m_w\|_2^2 + \text{Tr}\big(C + C_w - 2\big(CC_w\big)^{1/2}\big) \quad (4)$$

Mathematical Background and Notation: Consider a GAN generator G mapping from latent space with dimension n_z to an observation space with dimension n_x. We can define $Z := R^{n_z}$ and $X := R^{n_x}$ so that we may write $G : z \in Z \to x \in X$. At any $z \in Z$, G will have a Jacobian $J_z \in R^{n_x \times n_z}$ where $(J_z)_{i,j} := \frac{\partial G(z)_i}{\partial z_j}$. The object we care about will be the distribution of squared singular values of J_z. To see why, note that the mapping $M : z \in Z \to J_z^T J_z$ takes any point z to a symmetric and positive definite matrix of dimension $n_z \times n_z$ and so constitutes a Riemannian metric. We will write $J_z^T J_z$ as M_z (and refer to M_z somewhat sloppily as the "Metric Tensor"). If we know M_z for all $z \in Z$, we know most of the interesting things to know about the geometry of the manifold induced by G. In particular, fix some point $z \in Z$ and consider the eigenvalues $\lambda_1, \ldots, \lambda_{n_z}$ and eigenvectors $v_1, \ldots, v_{n_z}$ of M_z. Then for $\epsilon \in R$ and $k \in [1, n_z]$,

$$\lim_{\|\epsilon\| \to 0} \frac{\|G(z) - G(z + \epsilon v_k)\|}{\|\epsilon v_k\|} = \sqrt{\lambda_k} \quad (5)$$

Less formally, the eigenvectors corresponding to the large eigenvalues of M_z at some point $z \in Z$ give directions in which taking a very small "step" in Z will result in a large change in $G(z)$ (and analogously with the eigenvectors corresponding to the small eigenvalues). Because of this, many interesting things can be read out of the eigenspectrum of M_z.

Unfortunately, working with the whole spectrum is unwieldy, so it would be nicer to work with some summary quantity. In this work, we choose to study the condition number of M_z (the best justification we can give for this is that we noticed during exploratory analysis that the condition number was predictive of the Inception Score, but see the supplementary material for further justification of why we chose this quantity and not some other quantity). The condition number is defined for M_z as $\frac{\lambda_{max}}{\lambda_{min}}$. If the condition number is high, we say that the metric tensor is "poorly conditioned". If it's low, we say that the metric tensor is "well conditioned".

Now note that the eigenvalues of M_z are identical to the squared singular values of J_z. This is why we care about

the singular value spectrum of J_z.

3. Analyzing the Local Geometry of GAN Generators

The Metric Tensor Becomes Ill-Conditioned During Training: We fix a batch of $z \sim p(z)$ and examine the condition number of M_z at each of those points as a GAN is training on the MNIST data-set. A plot of the results is in Figure 2, where it can be seen that M_z starts off well-conditioned everywhere and quickly becomes poorly conditioned everywhere. There is considerable variance in how poor the conditioning is, with the log-condition-number ranging from around 12 to around 20.

It is natural to ask how consistent this behavior is across different training runs. To that end, we train 10 GANs that are identical up to random initialization and compute the average log-condition number across a fixed batch of z as training progresses (Figure 1 Top-Left). Roughly half of the time, the condition number increases rapidly and then stays high or climbs higher. The other half of the time, it increases rapidly and then decreases. This distribution of results is in keeping with the general understanding that GANs are "unstable".
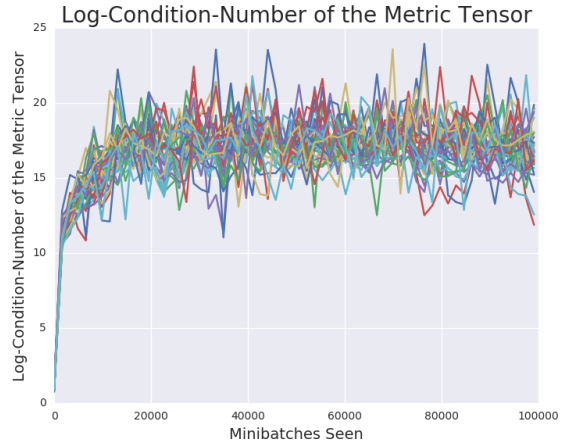

Figure 2. The condition number of M_z for a GAN trained on MNIST at various fixed z throughout training.

The condition number is informative and computing its average over many z gives us a single scalar quantity that we can evaluate over time. However, this is only one of many such quantities that we can compute, and it obscures certain facts about the singular value spectrum of J_z at various z. For completeness, we also compute – following Hoffman (2017), who does the same for variational autoencoders – the spectrum of the average (across a batch of z) Jacobian. It's not clear a priori what one should expect of these spec-
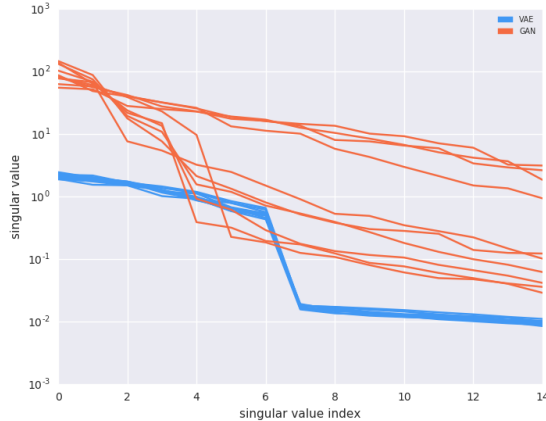
Figure 3. Log spectra of the average Jacobian from 10 training runs of a variational autoencoder and 10 training runs of a GAN. There are a few interesting things about this experiment: First, it gives a way to quantify how much less 'stable' the GAN training procedure is than the VAE training procedure. The spectra of the different VAE runs are almost indistinguishable. Second, though the GAN and VAE decoders both take noise from $N(0, I)$ as input, the overall sensitivity of the VAE decoder to its input seems to be quite a bit lower than that of the GAN decoder – this does not stop the VAE from successfully modeling the MNIST dataset.

tra, so to provide context we perform the same computation on 10 training runs of a variational autoencoder (Kingma & Welling, 2013; Rezende et al., 2014). See Figure 3 for more details. For convenience, we will largely deal with the condition number going forward.

Conditioning is Predictive of Other Quality Metrics: One reason to be interested in this condition number quantity is that it corresponds strongly to other metrics used to evaluate GAN quality.

We take two existing metrics for GAN performance and measure how they correspond to the average log condition number of the metric tensor. The first measure is the Inception Score (Salimans et al., 2016) and the second measure is the Frechet Inception Distance (Heusel et al., 2017). We test GANs trained on three datasets: MNIST, CIFAR-10, and STL-10 (LeCun et al., 1998; Krizhevsky, 2009; Coates et al., 2011). On the MNIST dataset, we modify both of these scores to use a pre-trained MNIST classifier rather than the Inception classifier. On the CIFAR-10 and STL-10 datasets, we use the scores as defined. We resized the STL-10 dataset to 48×48 as has become standard in the literature about GANs. The hyperparameters we use are those from Radford et al. (2015), except that we modified the generator where appropriate so that the output would be of the right size.

We first discuss results on the MNIST dataset. The left column of Figure 1 corresponds to (the same) 10 runs of the GAN training procedure with different random initializations. From top to bottom, the plots show the mean (across the latent space) log-condition number, the classifier score, and the MNIST Frechet Distance. The correspondence between condition number and score is quite strong in both cases. For both the Classifier Score and the Frechet Distance, the 4 runs with the lowest condition number also have the 4 best scores. They also have considerably lower intra-run score variance. **Note also that the dark purple run, which transitions over time from being in the ill-conditioned cluster to the well-conditioned cluster, also transitions between clusters in the score plots.** Examples such as this provide evidence for the significance of the correspondence.

We conducted the same experiment on the CIFAR-10 and STL-10 datasets. The results from these experiments can be seen in the left columns of Figure 12 and Figure 13 respectively. The correspondence between condition number and the other two scores is also strong for these datasets. The main difference is that the failure modes on the larger datasets are more dramatic — in some runs, the Inception Score never goes above 1. For both datasets, however, we can see examples of runs with middling performance according to the score that also have moderate ill-conditioning: **In the CIFAR-10 experiments, the light purple run has a score that is in betweeen the "good cluster" and the "bad cluster", and it also has a condition number that is between these clusters. In the STL-10 experiments, both the red and light purple runs exhibit this pattern.**

Should we be surprised by this correspondence? We claim that the answer is yes. Both the Frechet Inception Distance and the Inception Score are computed using a pre-trained neural network classifier. The average condition number is a first-order approximation of sensitivity (under the Euclidean metric) that makes no reference at all to this classifier.

Conditioning is Related to Missing Modes: Both of the scores aim to measure the extent to which the GAN is "missing modes". The Frechet Inception Distance arguably measures this in a more principled way than does the Inception Score, but both are designed with this pathology in mind. We might wonder whether the observed correspondence is partly due to a relationship between generator conditioning and the missing-mode-problem. As a coarse-grained way to test this, we performed the following computation: Using the same pre-trained MNIST classifier that was used to compute the scores in Figure 1, we drew 360 samples from each of the 10 models trained in that figure and examined the distribution over predicted
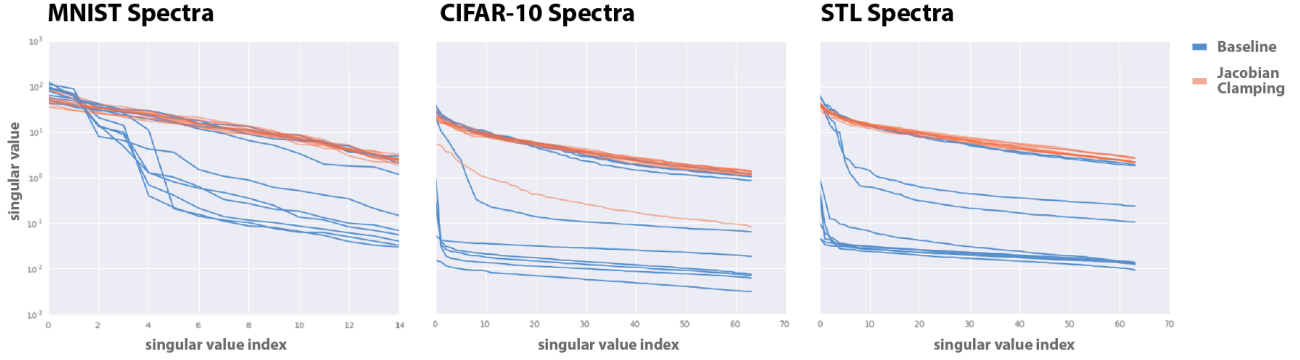
Figure 4. Singular value spectra of the average Jacobian at the end of training, presented in log-scale.

classes. We then found the class for which each model produced the fewest samples. The ill-conditioned models often had 0 samples from the least sampled class, and the well-conditioned models were close to uniformly distributed. In fact, the correlation coefficient between the mean log condition number for the model and the number of samples in the model's least sampled class was -0.86.

4. Jacobian Clamping

Given that the conditioning of J_z corresponds to the Inception Score and FID, it is natural to wonder if there is a causal relationship between these quantities. The notion of causality is slippery and causal inference is an active field of research (see Pearl (2009) and Woodward (2005) for overviews from the perspective of computer science and philosophy-of-science respectively) so we do not expect to be able to give a fully satisfactory answer to this question. However, we can perform one relatively popular method for inferring causality (Hagmayer et al., 2007; Eberhardt & Scheines, 2007), which is to do an intervention study. Specifically, we can attempt to control the conditioning directly and observe what happens to the relevant scores. In this section we propose a method for accomplishing this control and demonstrate that it both improves the mean scores and reduces variance of the scores across runs. We believe that this result represents an important step toward understanding the GAN training procedure.

Description of the Jacobian Clamping Technique: The technique we propose here is the simplest technique that we could get working. We tried other more complicated techniques, but they did not perform substantially better. An informal description is as follows: We feed 2 mini-batches at a time to the generator. One batch is noise sampled from p_z, the other is identical to the first but with small perturbations added. The size of the perturbations is governed by a hyperparameter ϵ. We then take the norm of the change in outputs from batch to batch and divide it by the norm

---

Algorithm 1 Jacobian Clamping

Input: norm ϵ, target quotients $\lambda_{max}, \lambda_{min}$, batch size B
repeat
    $z \in R^{B \times n_z} \sim p_z$.
    $\delta \in R^{B \times n_z} \sim N(0, 1)$.
    $\delta := (\delta / ||\delta||)\epsilon$
    $z' := z + \delta$
    $Q := ||G(z) - G(z')||/||z - z'||$
    $L_{max} = (\max(Q, \lambda_{max}) - \lambda_{max})^2$
    $L_{min} = (\min(Q, \lambda_{min}) - \lambda_{min})^2$
    $L = L_{max} + L_{min}$
    Perform normal GAN update on z with L added to generator loss.
until Training Finished

---

of the change in inputs from batch to batch and apply a penalty if that quotient becomes larger than some chosen hyperparameter λ_{max} or smaller than another hyperparameter λ_{min}. The rough effect of this technique should be to encourage all of the singular values of J_z to lie within $[\lambda_{min}, \lambda_{max}]$ for all z. See Algorithm 1 for a more formal description.

With respect to the goal of performing an intervention study, Jacobian Clamping is slightly flawed because it does not directly penalize the condition number. Unfortunately, directly penalizing the condition number during training is not straightforward due to issues efficiently estimating the smallest eigenvalue (Golub & Van Loan, 1996). We choose not to worry about this too much; We are more interested in understanding how the spectrum of J_z influences GAN training than in whether the condition number is precisely the right summary quantity to be thinking about.

Jacobian Clamping Improves Mean Score and Reduces Variance of Scores: In this section we evaluate the effects of using Jacobian Clamping. Our aim here is not

Baseline   **Jacobian Clamping**

The mean Log **Condition Number** of the metric tensor describes how consistently distance in the latent space maps to distance in the output space.

Frechet Distance is another standard metric of GAN performance. Lower is better.

We've left classifier score for the appendix because it's very similar to Frechet distance.
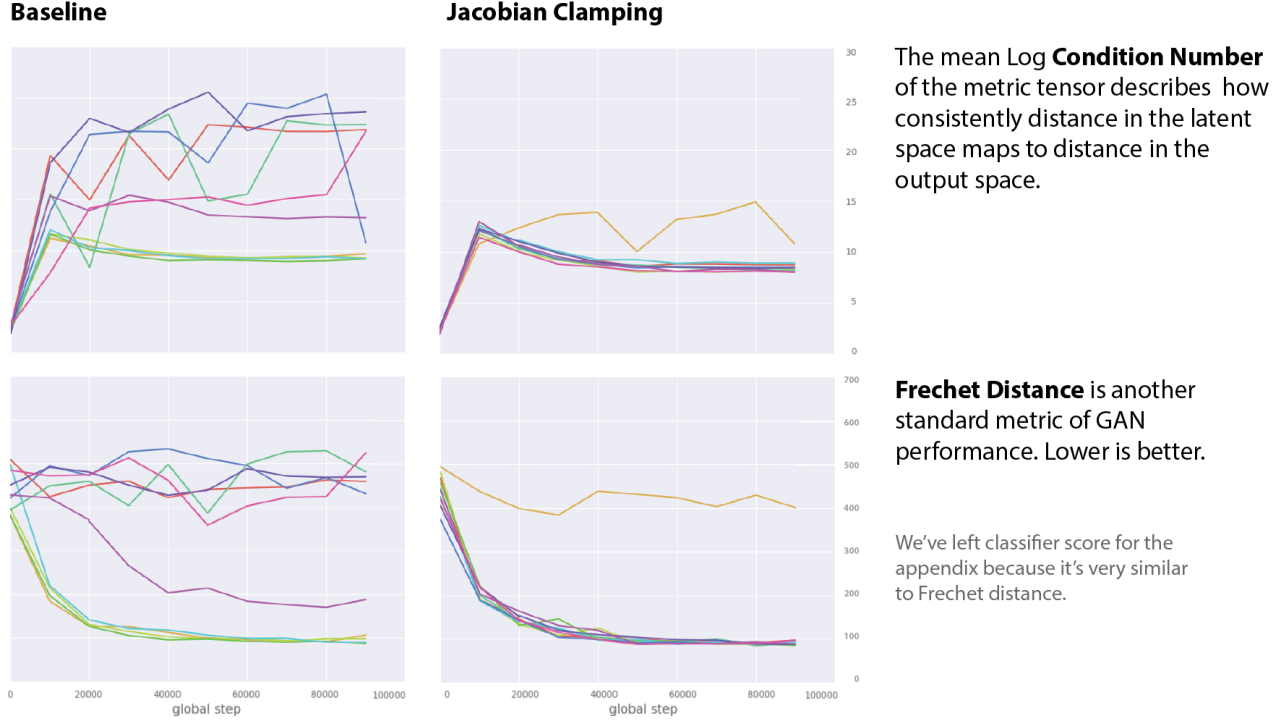
Figure 5. CIFAR10 Experiments. Left and right columns correspond to 10 runs without and with Jacobian Clamping, respectively. Within each column, each run has a unique color. Top Row: Mean log-condition number over time. Bottom Row: Frechet Inception Distance over time. Note the light purple run (Left) which has a condition number between the ill-conditioned cluster and the well-conditioned one; it also has scores between the low-scoring cluster and the high-scoring one. Note the gold run (Right): it's the only run for which Jacobian Clamping "failed", and it's also the only run for which the condition number did not decrease after its initial period of growth. We felt that there was little information conveyed by the Inception Score that was not conveyed by the Frechet Inception Distance, so for reasons of space we have put the Inception Score plots in the supplementary material.

to make claims of State-of-the-Art scores[4] but to provide evidence of a causal relationship between the spectrum of M_z and the scores. Jacobian Clamping directly controls the condition number of M_z. We show (across 3 standard datasets) that when we implement Jacobian Clamping, the condition number of the generator is decreased, and there is a corresponding improvement in the quality of the scores. This is evidence in favor of the hypothesis that ill-conditioning of M_z "causes" bad scores.

Specifically, we train the same models as from the previous section using Jacobian Clamping with a λ_{max} of 20, a λ_{min} of 1, and ϵ of 1 and hold everything else the same. As in the previous section, we conducted 10 training runs for each dataset. Broadly speaking, the effect of Jacobian Clamping was to prevent the GANs from falling into the ill-conditioned cluster. This improved the average case performance, but didn't improve the best case performance. For

---

[4] We regard these claims as problematic anyway. One issue (among many) is that scores are often reported from a single run, while the improvement in score associated with a given method tends to be of the same scale as the inter-run variance in scores.

all 3 datasets, we show terminal log spectra of $E_z[J_z]$ in Figure 4.

We first discuss the MNIST results. The right column of Figure 1 shows measurements from 10 runs using Jacobian Clamping. As compared to their "unregularized" counterparts in the left column, the runs using Jacobian Clamping all show condition numbers that stop growing early in training. The runs using Jacobian Clamping have scores similar to the best scores achieved by runs without. The scores also show lower intra-run variance for the "regularized runs".

The story is similar for CIFAR-10 and STL-10, the results for which can be seen in the right columns of Figures 12 and 13 respectively. For CIFAR-10, 9 out of 10 runs using Jacobian Clamping fell into the "good cluster". The run that scored poorly also had a generator with a high condition number. **It is noteworthy that the failure mode we observed was one in which the technique failed to constrain the quotient Q rather than one in which the quotient Q was constrained and failure occured anyway.** It is also (weak) evidence in favor of the causality hypothesis (in particular, it is evidence against the alternative hypoth-
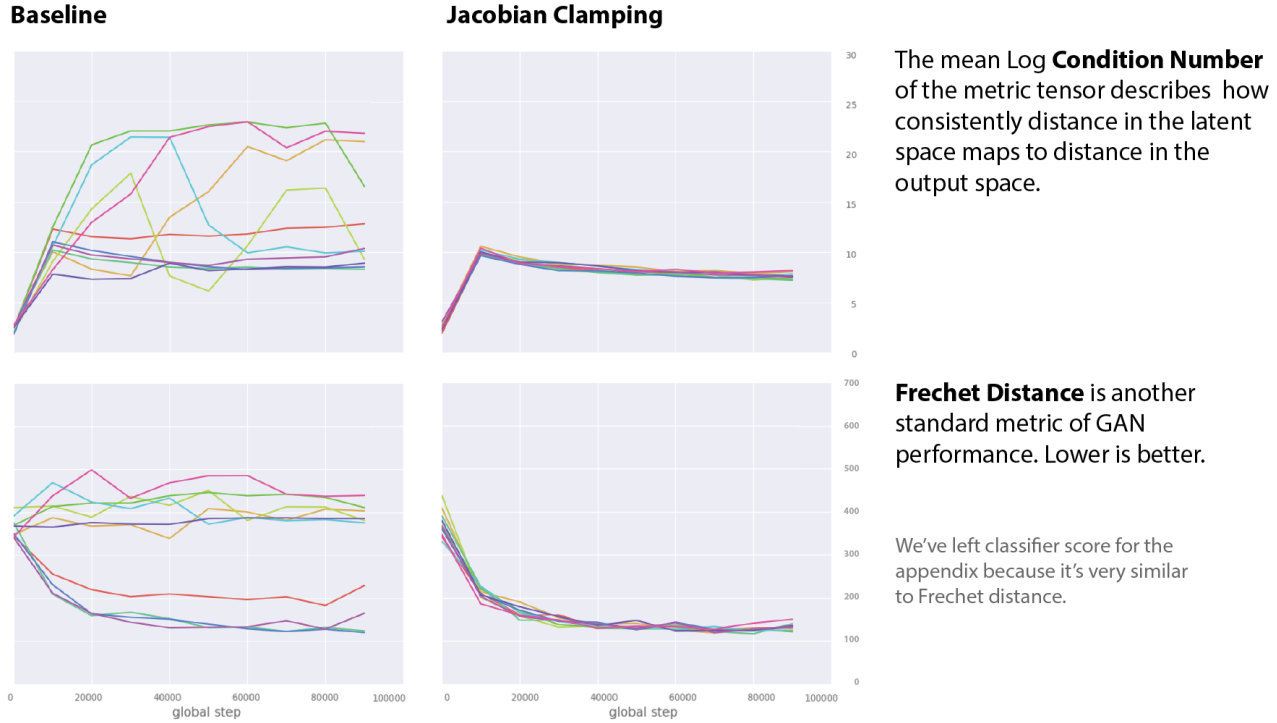
Baseline **Jacobian Clamping**

The mean Log **Condition Number** of the metric tensor describes how consistently distance in the latent space maps to distance in the output space.

Frechet Distance is another standard metric of GAN performance. Lower is better.

We've left classifier score for the appendix because it's very similar to Frechet distance.

Figure 6. STL10 Experiments. Left and right columns correspond to 10 runs without and with Jacobian Clamping, respectively. Within each column, each run has a unique color. Top Row: Mean log-condition number over time. Bottom Row: Frechet Inception Distance over time. Note the red run (Left): the generator has a condition number between the ill-conditioned cluster and the well-conditioned one; it also has scores between the low-scoring cluster and the high-scoring one. Note also the light purple run (Left) which is similar. As in Figure 12, we have moved the Inception Score plots to the supplementary material.

esis that Jacobian Clamping acts to increase scores in some other way than by constraining the conditioning). For STL-10, all runs fell into the good cluster.

It's worth mentioning how we chose the values of the hyperparameters: For ϵ we chose a value of 1 and never changed it because it seemed to work well enough. We then looked at the empirical value of the quotient Q from Algorithm 1 during training without Jacobian Clamping. We set $\lambda_{min}, \lambda_{max}$ such that the runs that achieved good scores had Q mostly lying between those two values. We consider the ability to perform this procedure an advantage of Jacobian Clamping. Most techniques that introduce hyperparameters don't come bundled with an algorithm to automatically set those hyperparameters.

We have observed that intervening to improve generator conditioning improves generator performance during GAN training. In the supplementary material, we discuss whether this relationship between conditioning and performance holds for all possible generators.

Jacobian Clamping Speeds Up State-of-the-Art Models: One limitation of the experimental results we've discussed so far is that they were obtained on a baseline model

that does not include modifications that have very recently become popular in the GAN literature. We would like to know how Jacobian Clamping interacts with such modifications as the Wasserstein loss (Arjovsky et al., 2017), the gradient penalty (Gulrajani et al., 2017), and various methods of conditioning the generator on label information (Mirza & Osindero, 2014; Odena et al., 2016). Exhaustively evaluating all of these combinations is outside the scope of this work, so we chose one existing implementation to assess the generality of our findings.

We use the software implementation of a conditional GAN with gradient penalty from `https://github.com/igul222/improved_wgan_training` as our baseline because this is the model from Gulrajani et al. (2017) that scored the highest. With its default hyperparameters this model has little variance in scores between runs but is quite slow, as it performs 5 discriminator updates per generator update. It would thus be desirable to find a way to achieve the same results with fewer discriminator updates. Loosely following Bellemare et al. (2017), we jointly vary the number of discriminator steps and whether Jacobian Clamping is applied. **Using the same hyperparameters as in previous experiments (that is, we made no attempt**
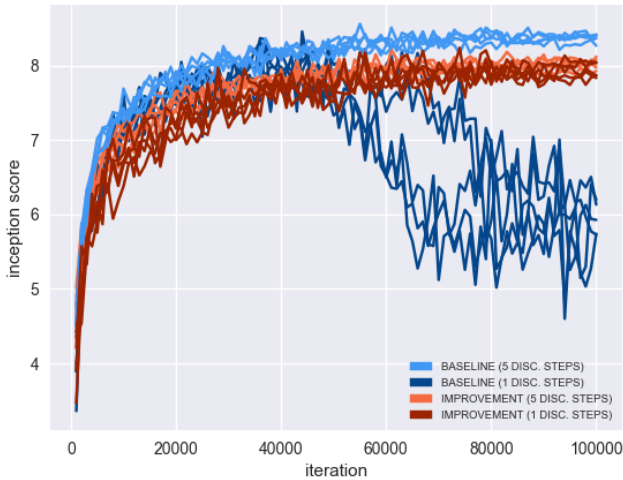
Figure 7. We train a class-conditional WGAN-GP under 4 different settings: With and without Jacobian Clamping and with 5 and 1 discriminator updates. We perform 5 trials for each setting. We made no effort whatsoever to tune our hyperparameters. We find that when using only 1 discriminator update the baseline model "collapses" but the model with Jacobian Clamping does not. When using 5 discriminator updates, the model with Jacobian Clamping performs slightly worse in terms of Inception Score, but this difference is small and could easily be due to the fact that we did not perform any tuning. When using Jacobian Clamping, reducing the number of discriminator steps does not reduce the score, but it more than halves the wall-clock time.

to tune for score) we find that reducing the number of discriminator updates and adding Jacobian Clamping more than halves the wall-clock time with little degradation in score. See Figure 7 for more details.

5. Related Work

GANs and other Deep Generative Models: There has been too much work on GANs to adequately survey it here, so we give an incomplete sketch: One strand attempts to scale GANs up to work on larger datasets of high resolution images with more variability (Denton et al., 2015; Radford et al., 2015; Odena et al., 2016; Zhang et al., 2016; Karras et al., 2017; Miyato & Koyama, 2018). Yet another focuses on applications such as image-to-image translation (Zhu et al., 2017), domain adaptation (Bousmalis et al., 2016), and super-resolution (Ledig et al., 2016). Other work focuses on addressing pathologies of the training procedure (Metz et al., 2016), on making theoretical claims (Arora et al., 2017) or on evaluating trained GANS (Arora & Zhang, 2017). In spectral normalization (Miyato et al., 2018), the largest singular value of the individual layer Jacobians in the discriminator is approximately penalized using the power method (see Golub & Van Loan (1996) for

an explanation of this). If Jacobian Clamping is performed with $\lambda_{min} = 0$, then it is vaguely similar to performing spectral normalization *on the generator*. See Goodfellow (2017) for a more full accounting.

Geometry and Neural Networks: Early work on geometry and neural networks includes the Contractive Autoencoder (Rifai et al., 2011) in which an autoencoder is modified by penalizing norm of the derivatives of its hidden units with respect to its input. Bengio et al. (2012) discuss an interpretation of representation learning as manifold learning. More recently, Kumar et al. (2017) improved semi-supervised learning results by enforcing geometric invariances on the classifier and Pennington et al. (2017) study the spectrum of squared singular values of the input-output Jacobian for feed-forward classifiers with random weights. In a related vein, three similar papers (Arvanitidis et al., 2017; Chen et al., 2017; Shao et al., 2017) have explicitly studied variational autoencoders through the lens of geometry.

Invertible Density Estimators and Adversarial Training: In Grover et al. (2017) and Danihelka et al. (2017), adversarial training is compared to maximum likelihood training of generative image models using an invertible decoder as in Dinh et al. (2014; 2016). They find that the decoder spectrum drops off more quickly when using adversarial training than when using maximum likelihood training. This finding is evidence that ill-conditioning of the generator is somehow fundamentally coupled with adversarial training techniques. Our work instead studies the variation of the conditioning among many runs of the same GAN training procedure, going on to show that this variation corresponds to the variation in scores and that intervening with Jacobian Clamping dramatically changes this variation. We also find that the ill-conditioning does not always happen for adversarial training — see Figure 4.

6. Conclusions and Future Work

We studied the dynamics of the generator Jacobian and found that (during training) it generally becomes ill-conditioned everywhere. We then noted a strong correspondence between the conditioning of the Jacobian and two quantitative metrics for evaluating GANs. By explicitly controlling the conditioning during training through a technique that we call Jacobian Clamping, we were able to improve the two other quantitative measures of GAN performance. We thus provided evidence that there is a causal relationship between the conditioning of GAN generators and the "quality" of the models represented by those GAN generators. We believe this work represents a significant step toward understanding GAN training dynamics.

Acknowledgements

References

Arjovsky, M. and Bottou, L. Towards Principled Methods for Training Generative Adversarial Networks. *ArXiv e-prints*, January 2017.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein GAN. *ArXiv e-prints*, January 2017.

Arora, S. and Zhang, Y. Do gans actually learn the distribution? an empirical study. *CoRR*, abs/1706.08224, 2017. URL http://arxiv.org/abs/1706.08224.

Arora, S., Ge, R., Liang, Y., Ma, T., and Zhang, Y. Generalization and equilibrium in generative adversarial nets (gans). *CoRR*, abs/1703.00573, 2017. URL http://arxiv.org/abs/1703.00573.

Arvanitidis, G., Hansen, L. K., and Hauberg, S. Latent Space Oddity: on the Curvature of Deep Generative Models. *ArXiv e-prints*, October 2017.

Bellemare, M. G., Danihelka, I., Dabney, W., Mohamed, S., Lakshminarayanan, B., Hoyer, S., and Munos, R. The cramer distance as a solution to biased wasserstein gradients. *CoRR*, abs/1705.10743, 2017. URL http://arxiv.org/abs/1705.10743.

Bengio, Y., Courville, A. C., and Vincent, P. Unsupervised feature learning and deep learning: A review and new perspectives. *CoRR*, abs/1206.5538, 2012. URL http://arxiv.org/abs/1206.5538.

Bousmalis, K., Silberman, N., Dohan, D., Erhan, D., and Krishnan, D. Unsupervised pixel-level domain adaptation with generative adversarial networks. *CoRR*, abs/1612.05424, 2016. URL http://arxiv.org/abs/1612.05424.

Chen, N., Klushyn, A., Kurle, R., Jiang, X., Bayer, J., and van der Smagt, P. Metrics for Deep Generative Models. *ArXiv e-prints*, November 2017.

Coates, A., Ng, A., and Lee, H. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 215–223, 2011.

Danihelka, I., Lakshminarayanan, B., Uria, B., Wierstra, D., and Dayan, P. Comparison of maximum likelihood and gan-based training of real nvps. *CoRR*, abs/1705.05263, 2017. URL http://arxiv.org/abs/1705.05263.

Denton, E. L., Chintala, S., Szlam, A., and Fergus, R. Deep generative image models using a laplacian pyramid of adversarial networks. *CoRR*, abs/1506.05751, 2015. URL http://arxiv.org/abs/1506.05751.

Dinh, L., Krueger, D., and Bengio, Y. NICE: nonlinear independent components estimation. *CoRR*, abs/1410.8516, 2014. URL http://arxiv.org/abs/1410.8516.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real NVP. *CoRR*, abs/1605.08803, 2016. URL http://arxiv.org/abs/1605.08803.

Eberhardt, F. and Scheines, R. Interventions and causal inference. *Philosophy of Science*, 74(5):981–995, 2007.

Golub, G. H. and Van Loan, C. F. *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996. ISBN 0-8018-5414-8.

Goodfellow, I. NIPS 2016 Tutorial: Generative Adversarial Networks. *ArXiv e-prints*, December 2017.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative Adversarial Networks. *ArXiv e-prints*, June 2014.

Grover, A., Dhar, M., and Ermon, S. Flow-gan: Bridging implicit and prescribed learning in generative models. *CoRR*, abs/1705.08868, 2017. URL http://arxiv.org/abs/1705.08868.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of wasserstein gans. *CoRR*, abs/1704.00028, 2017. URL http://arxiv.org/abs/1704.00028.

Hagmayer, Y., Sloman, S. A., Lagnado, D. A., and Waldmann, M. R. Causal reasoning through intervention. *Causal learning: Psychology, philosophy, and computation*, pp. 86–100, 2007.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. *ArXiv e-prints*, June 2017.

Hoffman, M. D. Learning deep latent Gaussian models with Markov chain Monte Carlo. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1510–1519, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR. URL http://proceedings.mlr.press/v70/hoffman17a.html.

Karras, T., Aila, T., Laine, S., and Lehtinen, J. Progressive Growing of GANs for Improved Quality, Stability, and Variation. *ArXiv e-prints*, October 2017.

Kingma, D. P. and Welling, M. Auto-Encoding Variational Bayes. *ArXiv e-prints*, December 2013.

Krizhevsky, A. Learning multiple layers of features from tiny images. 2009.

Kumar, A., Sattigeri, P., and Fletcher, P. T. Improved Semi-supervised Learning with GANs using Manifold Invariances. *ArXiv e-prints*, May 2017.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

Ledig, C., Theis, L., Huszar, F., Caballero, J., Aitken, A., Tejani, A., Totz, J., Wang, Z., and Shi, W. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. *ArXiv e-prints*, September 2016.

Metz, L., Poole, B., Pfau, D., and Sohl-Dickstein, J. Unrolled generative adversarial networks. *CoRR*, abs/1611.02163, 2016. URL http://arxiv.org/abs/1611.02163.

Mirza, M. and Osindero, S. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014. URL http://arxiv.org/abs/1411.1784.

Miyato, T. and Koyama, M. cGANs with projection discriminator. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ByS1VpgRZ.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1QRgziT-.

Nagarajan, V. and Kolter, J. Z. Gradient descent GAN optimization is locally stable. *CoRR*, abs/1706.04156, 2017. URL http://arxiv.org/abs/1706.04156.

Odena, A., Olah, C., and Shlens, J. Conditional Image Synthesis With Auxiliary Classifier GANs. *ArXiv e-prints*, October 2016.

Pearl, J. *Causality*. Cambridge university press, 2009.

Pennington, J., Schoenholz, S., and Ganguli, S. Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30*, pp. 4788–4798. Curran Associates, Inc., 2017.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2015. URL http://arxiv.org/abs/1511.06434.

Rezende, D., Mohamed, S., and Wierstra, D. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. *ArXiv e-prints*, January 2014.

Rifai, S., Vincent, P., Muller, X., Glorot, X., and Bengio, Y. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 833–840, 2011.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved Techniques for Training GANs. *ArXiv e-prints*, June 2016.

Shao, H., Kumar, A., and Fletcher, P. T. The Riemannian Geometry of Deep Generative Models. *ArXiv e-prints*, November 2017.

Woodward, J. *Making things happen: A theory of causal explanation*. Oxford university press, 2005.

Wu, Y., Burda, Y., Salakhutdinov, R., and Grosse, R. B. On the quantitative analysis of decoder-based generative models. *CoRR*, abs/1611.04273, 2016. URL http://arxiv.org/abs/1611.04273.

Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., and Metaxas, D. StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks. *ArXiv e-prints*, December 2016.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. *ArXiv e-prints*, March 2017.

A. Why Compute the Condition Number?

There are many summary statistics one could compute from the spectrum of the Jacobian. It is not obvious a priori that it makes sense to focus on the ratio of the maximum eigenvalue to the minimum eigenvalue, so here we make some attempt to justify that decision.

If you were to just glance at the spectra figures provided in the main text, using the log-determinant might seem like a reasonable thing to do. However, we note that (at least for the MNIST experiments) the largest singular values for the 'well behaved' runs are distinctly lower than those for the 'poorly behaved' ones. This suggests that the conditioning might be more pertinent than the determinant.

Even given that the conditioning is what's relevant, one could imagine other measures of Jacobian conditioning that less strongly emphasize the extreme singular values. Indeed, computing such quantities would be a useful exercise, and we expect that they would also correlate with GAN performance, but we have kept the condition number because it is simple and well-understood. We also feel that the condition number most closely relates to what is being optimized by the Jacobian Clamping procedure.

B. Additional Experimental Results

This section contains results that we have included for the purpose of completeness but which were not necessary for following the narrative of the paper. References to this section can be found in the main text.

B.1. Misbehaving Generators can be Well-Conditioned

We have observed that intervening to improve generator conditioning improves generator performance during GAN training. We also might like to know whether this relationship holds for all possible generators. Here we provide a counterexample of a deliberately pathological generator (not trained with a GAN loss) which is nonetheless well-conditioned. This suggests that the causal relationship we explore in the main text may relate to the GAN training process, and may not be an absolute property of generators in isolation.

We train a generator using the DCGAN architecture with a latent space of 64 dimensions. Rather than an adversarial loss, we train with an L2 reconstruction loss - in effect, teaching the generator to memorize the training examples it has seen. We select 10,000 examples to memorize: half of them (5,000) are random MNIST digits, and the other half are identical copies of a single MNIST sample (in this case, a four). We then establish a consistent but arbitrary mapping from 10,000 random z values to the training examples. The generator is trained with an L2 reconstruction loss to map each memorized z value to its associated training example. The generator's behavior on non-memorized z values is not considered at training time. There is no discriminator involved in this training procedure. Figure 8 shows the generator's output when provided the z values it was trained to associate with specific samples, indicating that it succeeds at memorizing the half-random half-identical data it was trained on.

At evaluation time, we provide random latent vectors, rather than the latent vectors the generator has been trained to memorize. Figure 9 shows the samples that this generator produces at evaluation time. This generator is clearly not well-behaved: it suffers from mode collapse (i.e. it often reproduces the single four that made up half of its training data) and mode dropping (i.e. even when it produces a novel sample, it usually looks an indistinct four or nine, and seldom looks like any other class). Figure 10 shows the label distribution as measured by a pre-trained classifier, confirming that this generator has a severe missing mode problem. This generator's poor behavior is also confirmed by its scores. Its Classifier Score is 4.95 for memorized z values and 2.22 on non-memorized z values. Its Frechet Distance is 118 for memorized z values and 240 for non-memorized z values.

Figure 11 shows that this poorly-behaved generator nonetheless has a good condition number. Taken in isolation, the trajectory of this generator's condition number would suggest that it belongs in the "good cluster" of Figure 1.

In summary, we demonstrate a generator that is *not* trained with a GAN loss, with conspicuous mode collapse and mode dropping, which is nonetheless well-conditioned. This suggests that the relationship between generator conditioning and generator performance does not hold for all generators, and suggests that it may instead be a property of GAN training dynamics.
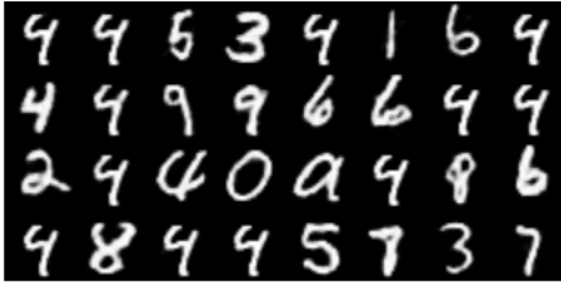
Figure 8. Samples from memorized zs. Half of the samples the generator was trained to memorize are identical copies of a single MNIST sample (in this case, a four) and the other half are random MNIST digits. The generator has successfully memorized the z-to-digit associations it was trained to reproduce.

Figure 9. Samples from random zs. The generator's behavior on these z values was not considered at training time. These samples often resemble the single four that made up half its training data, or other four- and nine-like digits. Occasionally, it produces indistinct digits that are not four-like, such as the 3 and the 5 in the bottom row, or indistinct samples that are not digit-like.
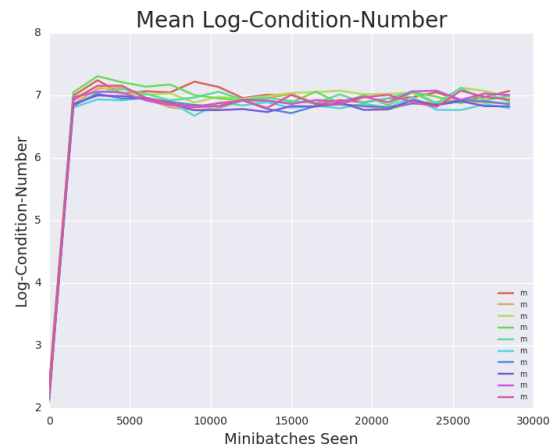

Figure 11. Mean log-condition number of misbehaving generator over 10 runs. Compare to Figure 1 in the main text: this misbehaving generator is better-conditioned than the "good cluster" of GAN generators.
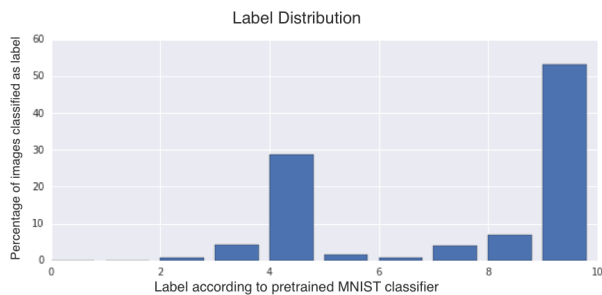
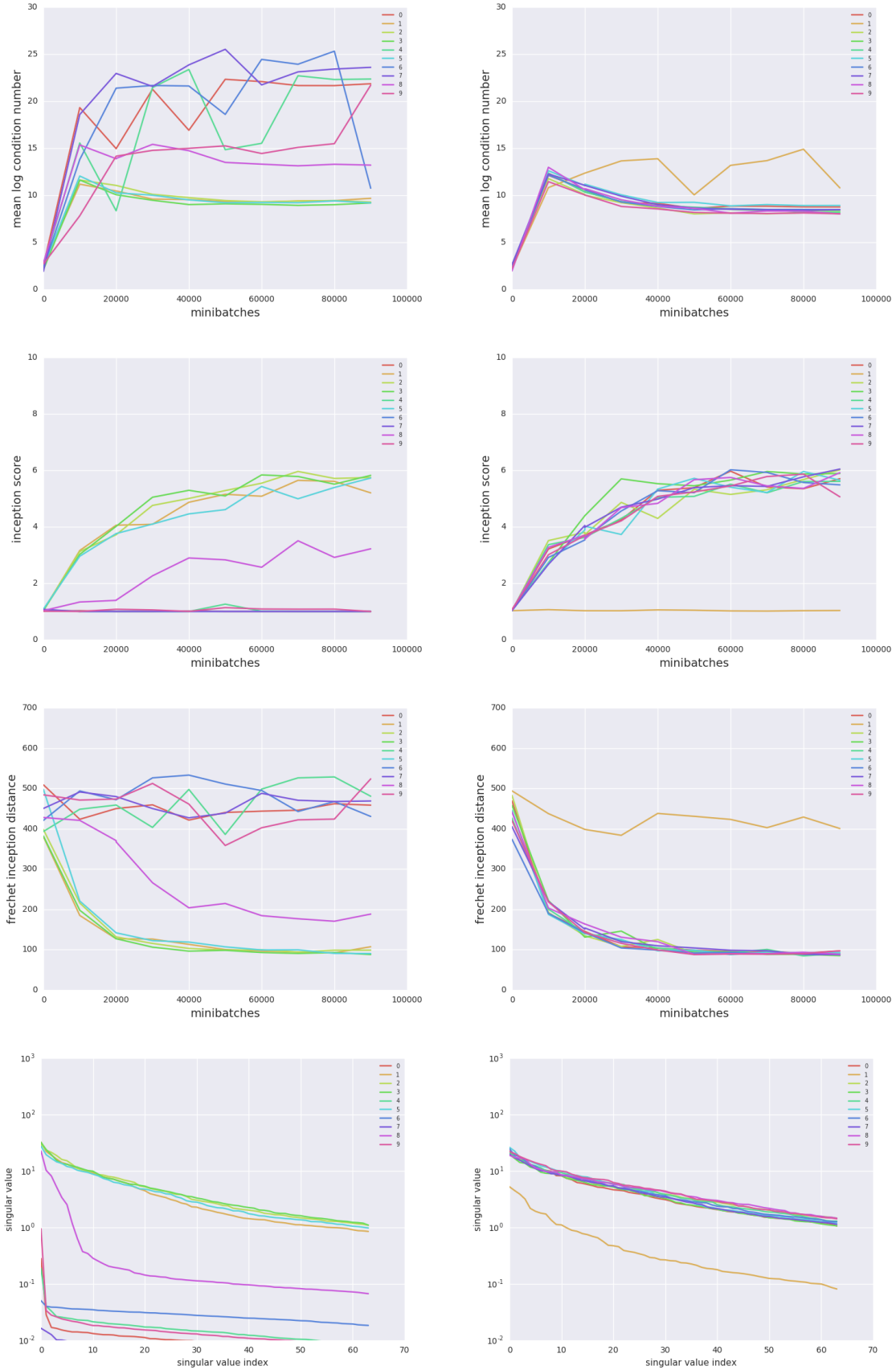*Figure 10.* Label distribution of samples from random zs

Figure 12. CIFAR10 Experimental results. Left and right columns correspond to 10 runs without and with Jacobian Clamping, respectively. Within each column, each run has a unique color.
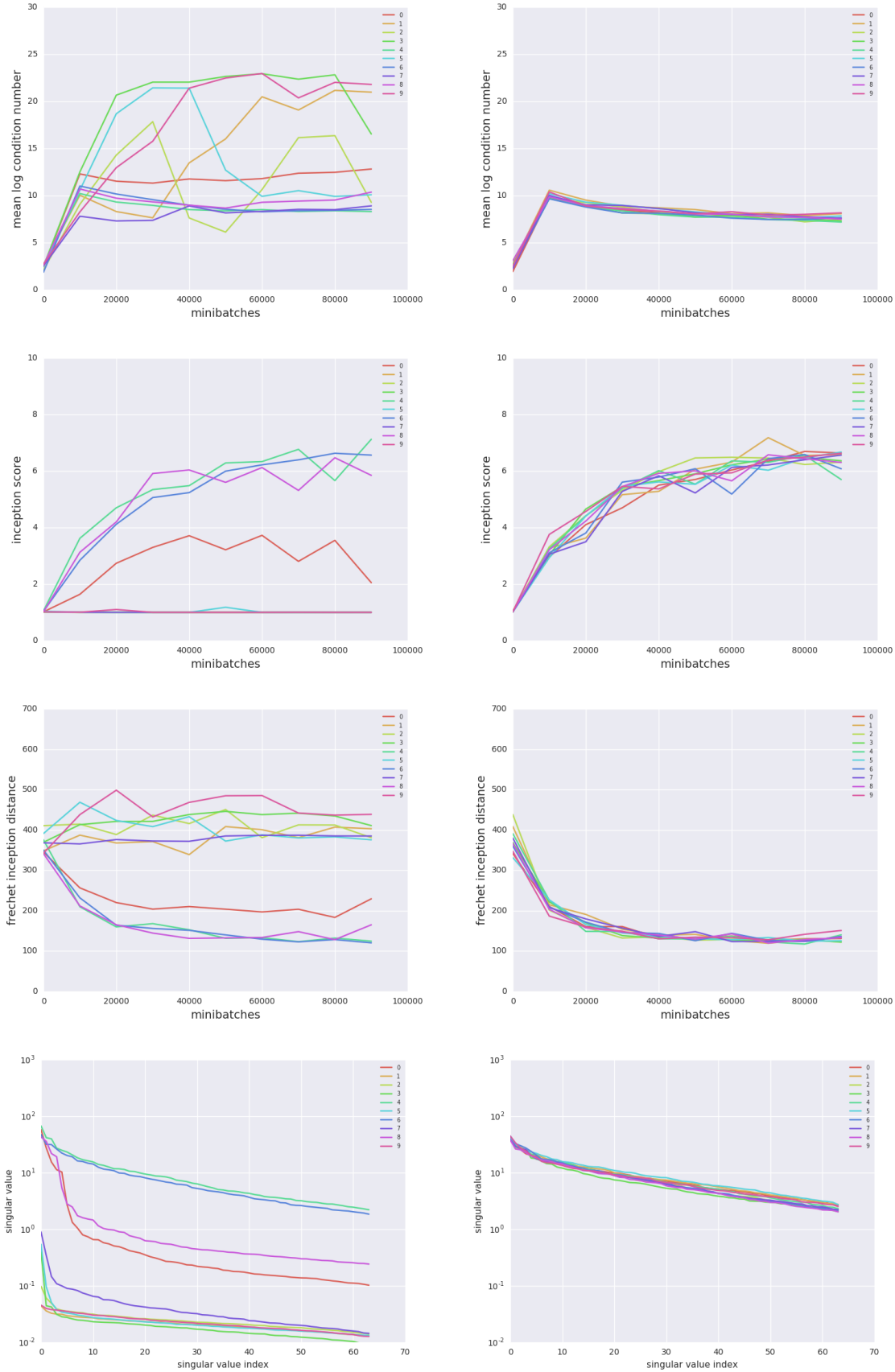
Figure 13. STL10 Experimental Results. Left and right columns correspond to 10 runs without and with Jacobian Clamping, respectively. Within each column, each run has a unique color.