GraN-GAN: Piecewise Gradient Normalization for Generative Adversarial Networks

Vineeth S. Bhaskara*1 Tristan Aumentado-Armstrong*1,2,3 Allan Jepson1 Alex Levinshtein1
1Samsung AI Centre Toronto 2University of Toronto 3Vector Institute for AI
{s.bhaskara,allan.jepson,alex.lev}@samsung.com, tristan.a@partner.samsung.com

Abstract

Modern generative adversarial networks (GANs) predominantly use piecewise linear activation functions in discriminators (or critics), including ReLU and LeakyReLU. Such models learn piecewise linear mappings, where each piece handles a subset of the input space, and the gradients per subset are piecewise constant. Under such a class of discriminator (or critic) functions, we present Gradient Normalization (GraN), a novel input-dependent normalization method, which guarantees a piecewise \( K \)-Lipschitz constraint in the input space. In contrast to spectral normalization, GraN does not constrain processing at the individual network layers, and, unlike gradient penalties, strictly enforces a piecewise Lipschitz constraint almost everywhere. Empirically, we demonstrate improved image generation performance across multiple datasets (incl. CIFAR-10/100, STL-10, LSUN bedrooms, and CelebA), GAN loss functions, and metrics. Further, we analyze altering the often untuned Lipschitz constant \( K \) in several standard GANs, not only attaining significant performance gains, but also finding connections between \( K \) and training dynamics, particularly in low-gradient loss plateaus, with the common Adam optimizer.

1. Introduction

Generative adversarial networks (GANs) [12] are a class of generative models that have been shown to be very effective, especially for unsupervised high-resolution image generation [27, 13, 16, 17]. GANs usually consist of two networks, a generator \( G(z) \) that generates synthetic data conditioned on a noise vector \( z \) (sampled from a known noise distribution, usually standard normal) and a discriminator (or critic) \( D(x) \) that classifies real data from the generated synthetic data. \( G \) and \( D \) are generally parametrized as deep neural networks and optimize a mini-max objective. A Nash equilibrium of the zero-sum game is attained when \( G \) models the real data distribution and \( D \) is maximally uncertain in discriminating the real from synthetic samples.

Despite the effectiveness of GANs in modeling high-dimensional data distributions, they are hard to train. The quality of the images output by \( G \) is dependent on the magnitude of the input gradients of the generator loss \( \mathcal{L}_G \), written

\[
\nabla_x \mathcal{L}_G(D(x)) = \nabla_D \mathcal{L}_G(D(x)) \nabla_x D(x),
\]

where \( x = G(z) \). The characteristics of the input gradient of \( \mathcal{L}_G \), in general, are determined both by the architecture of the discriminator \( D(x) \) as well as the loss function \( \mathcal{L}_G \) used in formulating the mini-max objective. In turn, \( \nabla_x D(x) \) is a function of the parameters \( \theta_D \) of the discriminator (or critic) that is trained to minimize a loss \( \mathcal{L}_D \) to separate real samples \( x_r \) from the synthetically generated ones \( x_f = G(z) \) for a given \( G \). Therefore, in contrast to image classification, the

![Figure 1: Images generated by WGAN-GP, SNGAN, and our method (GraND-GAN) on 128 × 128 LSUN-Bedrooms (zoom in for better viewing). Lower FID is better.](image-url)
role of $D$ is not only to accurately discriminate real data from fake or synthetic data, but also to have a well-behaved input gradient $\nabla_x D(x)$, which is the primary signal that $G$ relies on for learning. Designing such a discriminator is a major objective for GAN research.

Towards this goal, in this paper, we present a novel input-dependent normalization method called Gradient Normalization or GraN, which guarantees bounded gradients and a piecewise Lipschitz constraint almost everywhere. GraN can be applied to neural networks with piecewise linear activation functions, a prominent class of function approximators within deep learning, and we use the normalized output for discriminators (or critics). Fig. 1 shows a qualitative comparison of our method on the popular dataset LSUN-Bedrooms.

Our main contributions are as follows:

- We present Gradient Normalization (GraN) for piecewise linear networks $f(x)$ that strictly constrains $f$ to be piecewise $K$-Lipschitz, where the input space $x$ is partitioned into convex polytopes in each of which $f(x)$ is linear with $\|\nabla_x f(x)\| = K$.

- We show that normalizing discriminators (or critics) with GraN bounds the gradients received by the generator $G$ almost everywhere, stabilizing GAN training.

- Unlike spectral normalization (SN), GraN does not restrict processing at the individual layers and does not suffer from gradient attenuation. Further, in contrast to both SN and gradient penalties, GraN enforces the piecewise Lipschitz property as a hard constraint.

- Empirically, GraN performs better or competitive to existing methods on multiple datasets (CIFAR-10, CIFAR-100, STL-10, LSUN bedrooms, and CelebA), and two loss functions (discriminators with a non-saturating cross-entropy loss and critics with a soft hinge loss).

- While GraN only enforces a local, piecewise Lipschitz constraint, we find the finite-difference gradient norm is empirically well-behaved across large step sizes, likely including jumps across the polytopes modeled by $f$.

- We also investigate the effect of the Lipschitz constant $K$ on standard baseline models, finding (a) constrained discriminators (trained with cross-entropy loss) outperform constrained critics, (b) $K$ influences training dynamics when using Adam, especially on loss plateaus, and (c) tuning $K$ can significantly improve performance.

2. Related Work

2.1. Stabilizing GANs

Most previous works on stabilizing GANs take one of the following approaches: (1) proposing novel loss functions (e.g., [24, 4]), (2) devising improved architectures (e.g., [17, 39, 40, 28, 11, 9, 15]) or (3) introducing new constraints on $D$ (e.g., [2, 3, 31, 35]). Herein, we continue along the latter line of research, constructing an architectural regularization on $D$ that improves training without sacrificing network capacity.

One of the earliest works on constraining $D$ to stabilize GAN training is the Wasserstein GAN (WGAN) [3]. They propose a novel loss function and present weight-clipping as a way to regularize $D(x)$ to be 1-Lipschitz in $x$. However, subsequent work [13] showed gradient penalties to be more effective, as they do not impede optimization or severely reduce network capacity.

2.2. Gradient Penalties

Recent research has found gradient penalties (GPs), of the form $P_\delta(x) = (\|\nabla_x f(x)\|_2 - \delta)^2$, to be useful for GANs. Building on WGAN, among the most popular GAN variants is WGAN-GP [13], which forgoes weight clipping by applying $P_\delta$ along random convex combinations of reals and fakes, since any $C^0$ function with unit-length gradient is necessarily 1-Lipschitz. Later research considered one-sided GPs [30], as well as alternate sampling methods [36].

Part of the motivation for turning to optimal transport distances is the reduction of gradient uninformativeness, which is a problem for most $f$-divergence-based GANs when the real and fake distributions do not sufficiently overlap [31]. However, Zhou et al. [41] show that Lipschitz continuity can combat gradient uninformativeness in GANs more generally. Separately, Roth et al. [31] showed that increasing distributional overlap via noise is approximately equivalent to a zero-centered GP. This was simplified to the popular “R1” GP (defined as $P_0$ on real data) [25], used in recent state-of-the-art GANs (e.g., [17, 7]).

Clearly, gradient regularization has seen empirical success in improving GANs. Yet, a downside of soft GPs is that they may not enforce exactly the desired value at a given position; furthermore, they are only applied to a subset of the input domain, which may shift over time. In contrast, GraN enforces unit gradients almost everywhere by construction.
2.3. Weight Normalization

Weight normalization (WN) [33] reparametrizes layers in a manner conducive to better conditioned optimization, used in early GAN work [32]. In WN, for each individual layer $i$ of the network, the corresponding weight vectors are rewritten as $\tilde{w}_i = w_i \rho_i / ||w_i|| \forall i$, where the learned scalar $\rho_i$ controls the norm and $w_i / ||w_i||$ represents the direction.

For piecewise linear networks, each $x$ is linearly mapped to $f(x) = w \cdot x + b$, where $w$ is locally constant around $x$, is implicitly input-dependent, and may be interpreted as the effective weight vector of a (local) linear model (see §3.2 for details). In a manner reminiscent of WN, GraN essentially normalizes $f$ by the norm of its gradient, i.e., $\nabla x f = ||w||$. In contrast, however, GraN acts on the full network, rather than a single layer at a time, and enforces piecewise constant gradients, rather than reparametrizing the network.

2.4. Spectral Normalization and Gradient Attenuation under Global Lipschitz Constraints

Building on prior regularizations, such as GPs and weight clipping, Miyato et al. [27] present spectral normalization (SN) as an alternative method of ensuring $D$ is 1-Lipschitz, without an additional penalty in the objective for $D$. This is enforced by a layer-wise weight normalization technique, dividing by an estimate of the maximal singular value from each weight matrix. Empirically, SN is an effective stabilizer for GANs during training, independent of the loss function employed. As a result, it is a major component in recent large-scale GANs [6, 34]. We discuss SN further in §5.

At an architectural level, balancing network capacity and regularization is difficult for globally Lipschitz-constrained networks. Indeed, Anil et al. [1] showed that standard networks struggle to solve simple tasks when globally Lipschitz constrained, and that smooth SNed ReLU networks with unit gradients become globally linear, which they solve by enforcing gradient norm preservation and weight matrix orthnormality. We avoid this in GraN by permitting discontinuities in $D$ with respect to the input space and by only constraining the Lipschitz constant $K$ locally in a piecewise manner. Later work [21] combated the gradient attenuation induced by Lipschitz constraints with a novel orthogonal convolution operator. In contrast, GraN can be applied on top of any piecewise linear network without globally constraining $K$, while doing so locally in a piecewise sense.

3. Background

3.1. Generative adversarial networks

Let $P_r$ represent the distribution of real data and $P_g$ be the distribution of generated data at a given state of the generator $G$. Let $L_G$ and $L_D$ represent the loss functions for the generator $G$ and the discriminator (or critic) $D$, respectively. Let $z \sim \mathcal{N}(0, 1)$ be a $|Z|$-dimensional noise vector sampled from an i.i.d. standard normal distribution. Let $f : \mathbb{R}^d \rightarrow \mathbb{R}$ represent a deep neural network encoding a scalar field. For image generation, $\mathbb{R}^d = \mathbb{R}^{3 \times H \times W}$.

Goodfellow et al. [12] originally propose a cross-entropy (CE) loss-based objective for training $G$ and $D$ as follows:

$$L_D = \mathbb{E}_{x \sim P_r} [- \log D(x)] + \mathbb{E}_{x \sim P_g} [- \log (1 - D(x))],$$  \hspace{1cm} (2)

$$L_G = \mathbb{E}_{x \sim P_g} [\log (1 - D(x))],$$  \hspace{1cm} (3)

where the discriminator $D(x) = \sigma(f(x)) \in [0, 1]$ represents the probability of a sample $x$ coming from the real distribution $P_r$ and $\sigma(\cdot)$ is the sigmoid function $1/(1 + e^{-\cdot})$. Hence, $f$ computes a logit mapped to a probability by $\sigma(\cdot)$. The gradient of $L_G$ with respect to the inputs $x$ is then

$$\nabla_x L_G = \mathbb{E}_{x \sim P_g} [- D(x) \nabla_x f(x)].$$  \hspace{1cm} (4)

Notice that, when $D(x) \rightarrow 0$ for some generated $x \sim P_g$, the contribution to $\nabla_x L_G$ from such points will necessarily be small and have little influence on updating $G$. This is particularly probable early in training, when $P_g$ and $P_r$ are easily separated. To overcome this problem, in the same work, the authors suggest using an alternative objective for $G$ that is non-saturating, given as

$$L_G = \mathbb{E}_{x \sim P_g} [- \log D(x)],$$  \hspace{1cm} (5)

for which the gradient is

$$\nabla_x L_G = \mathbb{E}_{x \sim P_g} [-(1 - D(x)) \nabla_x f(x)].$$  \hspace{1cm} (6)

In this case, note that when $D$ confidently rejects fakes (i.e., $D(G(z)) \approx 0$), they can still contribute to $\nabla_x L_G$. For future convenience of notation, we use “NSGAN” to refer to the non-saturating (NS) version of training a discriminator via a GAN, where $L_D$ is given by Eq. (2) and $L_G$ is given by Eq. (5). Note that although $L_G$ in Eq. (5) is relatively non-saturating early in the training compared to Eq. (3), it may still saturate later in training as $P_g$ gets closer to $P_r$.

Subsequently, the WGAN model [3] attempted to address two issues with the original GAN formulation. First, due to the use of the Jensen-Shannon divergence, disjoint support two issues with the original GAN formulation. First, due to the use of the Jensen-Shannon divergence, disjoint support
where $D(x) = f(x)$, and $f(x)$ is constrained to be 1-Lipschitz in $x$, denoted by $\|f\|_{\text{Lip}} = 1$. We now have

$$
\nabla_x \mathcal{L}_G = \mathbb{E}_{x \sim P_g} \left[-\nabla_x f(x)\right],
$$

(9)

which does not saturate, unlike NSGAN (Eq. (6)).

Another commonly used variant of the critic loss $\mathcal{L}_D$ is the hinge loss [22, 27]:

$$
\mathcal{L}_D = \mathbb{E}_{x \sim P_x} \left[\text{ReLU}(1 - D(x))\right] + \mathbb{E}_{x \sim P_g} \left[\text{ReLU}(1 + D(x))\right],
$$

(10)

where $\text{ReLU}(x) = \max\{0, x\}$ is the rectified linear unit. It is straightforward to define a smooth analogue of the hinge loss in Eq. (10) by using a smooth approximation of ReLU defined as $\text{SoftPlus}(x) = \log(1 + e^x)$, which we refer to as the soft hinge loss. For convenience of notation, we call $D$ a critic with $D(x) = f(x)$, when $\mathcal{L}_D$ is any of the Wasserstein, hinge, or soft hinge loss, and a discriminator with $D(x) = \sigma(f(x))$, when $\mathcal{L}_D$ is a variant of the cross-entropy loss (as in NSGANs).

For GANs, several methods have been proposed to effectively constrain $D$, generally to 1-Lipschitz function spaces. WGAN uses weight clipping, which compromises optimization ease and capacity, while WGAN-GP imposes a soft gradient penalty to improve upon this (see §2.1 and §2.2). Present gradient normalization for discriminators and critics.

3.2. Deep piecewise linear networks

Modern deep neural networks predominantly use piecewise linear activation functions such as ReLU and LeakyReLU. Such activation functions do not admit regions with saturation, and, hence, allow training very deep networks effectively without vanishing gradients.

Let $f : \mathbb{R}^d \rightarrow \mathbb{R}$ represent a (deterministic) deep neural network with piecewise linear activation functions. Denote the network parameters by $\theta$. Then one may write

$$
f(x, \theta) = w(x, \theta) \cdot x + b(x, \theta),
$$

(13)

where $w(x, \theta)$ and $b(x, \theta)$ are piecewise constant in $x$. We denote $w(x, \theta)$ and $b(x, \theta)$ as the scalar dot-product of flattened tensors. Thus, $w(x, \theta)$ and $b(x, \theta)$ being piecewise constant in $x$ means that $\exists \text{ input sub-sets } S_k \in \{ S_j \}_j \subseteq \mathbb{R}^d$, where

$$
w(x \in S_k, \theta) = w_k(\theta) \text{ and } b(x \in S_k, \theta) = b_k(\theta),
$$

(14)

such that $w_k$ and $b_k$ are independent of $x$ within the sub-set $S_k$. Hence, $\forall x \in S_k$, we have

$$
f(x \in S_k, \theta) = w_k(\theta) \cdot x + b_k(\theta),
$$

(14)

which is linear in $x \in S_k$. Then $f(x, \theta)$ is the composition of continuous, piecewise linear functions, and is therefore itself a continuous and piecewise linear function of $x$. That is, there exist disjoint open input subsets $S_k$, such that $\cup_{k=1}^K S_k = \mathbb{R}^d$, where Eq. 14 holds. I.e., $f(x, \theta)$ is a linear function of $x \in S_k$, with coefficients $w_k$ and $b_k$ that only depend on $\theta$. One can interpret $w_k(\theta)$ and $b_k(\theta)$ as the effective weights and bias of a linear functional (given by Eq. (14)) that equals the predictions of the deep neural network, $f(x)$, $\forall x \in S_k$. Note that, unlike a linear hyperplane in logistic regression, the effective weights $w_k(\theta)$ and bias $b_k(\theta)$ are only applicable for $x \in S_k$.

Given this structure, for points off of $\partial S_k$ (the boundary of $S_k$), the gradient takes a simple form:

$$
\nabla_x f(x, \theta) = w_k(\theta),
$$

(15)

which is a constant vector $\forall x \in S_k$.

4. Gradient Normalization

In this section, we present gradient normalization (GraN), which strictly constrains piecewise linear networks to be 1-Lipschitz almost everywhere.
As in §3.2, let \( f(x) : \mathbb{R}^d \to \mathbb{R} \) represent a piecewise linear neural network with parameters \( \theta \). We then define the gradient normalized function \( g(x) \) as

\[
    g(x) = f(x) \mathcal{R}_c(||\nabla_x f(x)||) = \frac{f(x) \|| \nabla_x f(x) \||}{|| \nabla_x f(x) \||^2 + \epsilon} 
\]

(16)

where \( \epsilon > 0 \) is a fixed constant for numerical stability and \( \mathcal{R}_c(n) = n/(n^2 + \epsilon) \) is the normalization factor, with \( n = || \nabla_x f(x) || \). Any bounded \( R(n) \), with \( R(n) = (1/n)(1 + o(1)) \) as \( n \to \infty \), could be tried. We briefly experimented with \( R(n) = 1/(n + \epsilon) \), which produced similar although slightly worse results than \( \mathcal{R}_c \). The choice of \( R(n) \) could benefit from further study. We remark that a concurrent work [37] to ours independently explores a similar technique to regularize \( D \), but with a different normalization factor.

Given this normalization, if \( f \) is an arbitrary piecewise linear function then \( g \) is piecewise linear such that \( || \nabla_x g(x) || \leq 1 \) analytically almost everywhere in \( x \in \mathbb{R}^d \). While the gradient \( \nabla_x g(x) \) is bounded, \( g(x) \) itself can still take real values with no bounds, i.e., \( g(x) \in \mathbb{R} \).

One can better describe this result with the notation developed in §3.2. Consider an arbitrary input \( x \in S_k \) (without any loss of generality) that belongs to the input partial subset \( S_k \) and is mapped by the network to a linear piece given by \( f(x) = w_k(\theta) \cdot x + b_k(\theta) \forall x \in S_k \). Then one has \( \nabla_x f(x, \theta) = w_k(\theta) \forall x \in S_k \) (Eq. (15)). Therefore, the GraNed function \( g(x) \), given \( f(x) \), for \( x \in S_k \) becomes

\[
    g(x \in S_k) = [w_k(\theta) \cdot x + b_k(\theta)] \frac{||w_k(\theta)||}{||w_k(\theta)||^2 + \epsilon}, \quad (17)
\]

Consequently, \( \nabla_x g \) and its norm can be written as

\[
    \nabla_x g(x \in S_k) = w_k(\theta) \frac{||w_k(\theta)||}{||w_k(\theta)||^2 + \epsilon}, \quad (18)
\]

\[
    \implies || \nabla_x g(x \in S_k) || = \frac{||w_k(\theta)||^2}{||w_k(\theta)||^2 + \epsilon} < 1. \quad (19)
\]

Since \( x \) and \( S_k \) were arbitrarily chosen, it follows that \( || \nabla_x g(x) || < 1 \) except at the boundaries, say \( x \in \partial S_k \), where the gradient does not exist. Since \( \cup_k \partial S_k \) is measure zero, we have \( || \nabla_x g(x) || < 1 \) almost everywhere in \( \mathbb{R}^d \). Further, for \( ||w_k(\theta)|| \gg \epsilon \), we have \( || \nabla_x g || \approx 1 \) in \( S_k \).

Note that, since the original piecewise linear function \( f(x) \) does not have a smooth gradient \( \nabla_x f(x) \), the normalization factor \( \mathcal{R}_c(|| \nabla_x f(x)||) \) will have discontinuities for \( x \in \cup_k \partial S_k \). Therefore, \( g(x) \) is typically discontinuous and not guaranteed to be globally 1-Lipschitz. However, we empirically find that \( g(x) \) has bounded finite-differences over substantial perturbations (see §7).

Due to this piecewise constant gradient property (with unit-bounded norm), we remark that \( g \) is piecewise 1-Lipschitz, since \( g \) is 1-Lipschitz with respect to any pair of points within each subset \( S_k \). It is also locally 1-Lipschitz continuous almost everywhere, since there exists an open ball around every point \( x \in \cup_k S_k \), within which \( 1 \)-Lipschitz continuity holds. Empirically, we find that the tight bound on \( || \nabla_x g(x) || \) almost everywhere assists with GAN training.

### 4.1. Gradient Normalized GANs

Given a deep neural network represented by \( f(x) \), we write \( D(x) = \sigma(f(x)) \) when \( D \) represents discriminators, and \( D(x) = f(x) \), when \( D \) represents critics, respectively, where \( \sigma(\cdot) \) is the sigmoid function (see §3.1). When \( f(x) \) is piecewise linear in \( x \), we define the gradient normalized discriminator (GraND) and critic (GraNC) as \( D(x) = \sigma(g(x)) \) and \( D(x) = g(x) \), respectively, where

\[
    g(x) = \frac{f(x)}{\tau} \frac{|| \nabla_x f(x) ||}{|| \nabla_x f(x) ||^2 + \epsilon}, \quad (20)
\]

and \( \tau \) is a positive constant that constrains \( g \) to be \( K \)-Lipschitz, with \( K = 1/\tau \). For GraNDs, \( \tau \) takes a role analogous to the temperature of a sigmoid, and, hence, we term it as the “temperature” hyperparameter.

Like spectral normalization [27], the Lipschitz constant \( K = 1/\tau \) is the only additional hyperparameter that needs to be tuned for our method. Moreover, in practice, our method achieves a tight bound on the Lipschitz constant, unlike spectral normalization, which imposes a loose upper bound with \( ||g||_{\text{Lip}} \leq K \). See §5 for details.

Finally, note that since the gradients in Eqs. (11) and (12) are computed by back-propagation, the step discontinuities in \( g(x) \) are ignored. Therefore, for GANs with GraNds and GraNCs, the generator \( G \) receives gradients that are always bounded, i.e., \( || \nabla_x \mathcal{L}_G || \leq K \).

### 5. Comparison to Layer-wise Norms

**Spectral Normalization** As noted in §2.2, GPs softly encourage Lipschitz continuity in a data-dependent manner. Improving on this, the spectrally normalized GAN (SNGAN) [27] achieves it at the architectural level. In particular, a linear layer \( f(x) = Wx \) (ignoring the bias term) with weights \( W \) is normalized via \( \tilde{W} = W/||W||_2 \), before being applied to an input \( \tilde{x} = \tilde{W}x \), where \( ||W||_2 = \sigma_1(W) \) is the spectral norm of \( W \), equal to its largest singular value (SV) \( \sigma_1 \). This is a form of WN, but acts on the whole matrix rather than its individual rows. Notice that \( ||\tilde{f}||_{\text{Lip}} \leq \sigma_1(W) \) and \( \sigma_1(W) = 1 \), so \( ||\tilde{f}||_{\text{Lip}} \leq 1 \). As such, SN ensures layer-wise 1-Lipschitz continuity, and thus enforces it across the whole network, where SVs are estimated via power iteration.\(^1\)

One downside of SN is the tendency to over-constrain the network, reducing capacity and attenuating gradients, due to the layer-wise enforcement mechanism [1, 21]. SN guarantees a function is globally 1-Lipschitz by bounding the

---

\(^1\)Although note that, at each step, power iteration provides a lower bound on \( \sigma_1(W) \), and thus in practice it is possible that \( ||\tilde{f}||_{\text{Lip}} > 1 \).
Lipschitz constant (LC) of every layer, as this then bounds their composition: $||f \circ g||_{lip} \leq ||f||_{lip} ||g||_{lip}$. However, this upper-bound is often loose, over-constraining the network and attenuating gradients (expanded upon below). In comparison, GraN acts upon the network as a whole, leaving weights per layer free to vary, and ensuring that gradients with respect to the input are always of unit norm.

The Looseness of Layerwise Constraints For illustration, consider the simple case of two SNed linear layers without biases, ignoring non-linear activations (though, for instance, this occurs in the positive domain of ReLU): $z = f(g(x))$, where $f(y) = By$ and $g(x) = Ax$. Assuming sufficient power iterations under SN, the largest singular values (SVs) of $A$ and $B$ are one, assigning each layer an LC of one. We next examine the conditions for which the composition of the layers, $f \circ g$, also has a LC of one.

In this case, $z = f(g(x)) = BAx$, and so $f \circ g$ has a sharp LC of one if and only if the maximal SV of $BA$, namely $\sigma_1(BA)$, is also one. Let $\Gamma_A(A)$ denote the span of the right singular vectors of $A$ with corresponding SVs equal to $\sigma$. We show in supplementary material, §D, that $\sigma_1(BA) = 1$ if and only if the first principal angle $[10, 42]$ between the subspaces $\Gamma_A(A^T)$ and $\Gamma_A(B)$ is zero. This only occurs if they intersect in at least one dimension. However, if even one SV of $A$ and $B$ is less than one, then $\Gamma_A(A^T)$ and $\Gamma_A(B)$ will be measure zero, meaning the network must solve a high dimensional “alignment” problem of two measure zero sets. The only scenario avoiding this is when every SV of either $A$ or $B$ are one, which is also a measure zero event.

Importantly, since the SN framework does not directly encourage these subspaces to align, or all the SVs of the weight matrices to be one, in practice it is likely that $\sigma_1(BA) < 1$. This issue is exacerbated for deeper networks, as the overall LC equals the product of $\sigma_1(BA)$ for every adjacent pair of layers. In addition, though training may encourage the network to utilize its capacity by avoiding small SVs, empirically it struggles to do so [1]. In contrast, GraN not only guarantees the function is locally 1-Lipschitz, but does so without constraining individual layers (avoiding subspace alignment issues) and enforces exactly unit gradient almost everywhere as well (attaining the sharp LC bound within every $S_k$). Fig. 3 displays this exactness for GraN; note that SNGAN, due its residual architecture, actually has an LC of 1024, showcasing the looseness of the bound.

6. Experiments

Model Architectures and Training We evaluate our method on unconditional image generation across datasets of various sizes: CIFAR-10/100 (32 × 32) [19], STL-10 (48 × 48) [8], LSUN bedrooms (128 × 128) [38], and CelebA (128 × 128) [23]. We use Mimicry [20] with PyTorch [29] on a single NVIDIA V100 GPU for training our models. The generator $G$ and discriminator (or critic) $D$ architectures are identical across methods for a given dataset with identical number of learnable parameters for fair comparison. We use the Adam [18] optimizer with $\beta_1 = 0.0, \beta_2 = 0.9$, and a batch size of 64 for 100K iterations, with a dataset-dependent learning rate (LR) $\alpha$ and number of $D$ steps per $G$ step $n_{dis}$. NSGANs and GraND-GAN use the cross-entropy loss for $D$ in Eq. (2) and the non-saturating loss for $G$ in Eq. (5). GraNC-GAN uses a soft version of the hinge loss for $D$ in Eq. (10), replacing ReLU with softplus, and Eq. (8) for $G$ (see §3.1). The soft hinge loss was found to be more performant and stable than the (hard) hinge loss with GraNC-GANs: on LSUN, GraNC diverged with the hard hinge loss, while on CelebA, a lower FID was obtained. See supplementary material for further model details, ablation experiments, and hyperparameter choices.

Table 1: Hyperparameters tested in Fig. 2 for GraND-GAN, SNGAN, and WGAN-GP on CIFAR-10, where $\alpha$ is the learning rate, $\beta_1$ and $\beta_2$ parametrize Adam in Eq. (21), and $n_{dis}$ is the number of discriminator steps per generator step.

<table>
<thead>
<tr>
<th>Setting</th>
<th>$\alpha$ (LR)</th>
<th>$\beta_1$</th>
<th>$\beta_2$</th>
<th>$n_{dis}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.0001</td>
<td>0.5</td>
<td>0.9</td>
<td>5</td>
</tr>
<tr>
<td>B</td>
<td>0.0002</td>
<td>0.5</td>
<td>0.999</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>0.001</td>
<td>0.5</td>
<td>0.999</td>
<td>5</td>
</tr>
<tr>
<td>D</td>
<td>0.001</td>
<td>0.9</td>
<td>0.999</td>
<td>5</td>
</tr>
<tr>
<td>E (default)</td>
<td>0.0002</td>
<td>0.0</td>
<td>0.9</td>
<td>5</td>
</tr>
</tbody>
</table>

Figure 2: Inception scores (IS), FIDs, and KIDs on CIFAR-10 image generation across different hyperparameters listed in Table 1 for GraND-GAN, WGAN-GP, and SNGAN. Results show superior robustness for GraND.

Figure 3: Boxplots of gradient norms at real (blue) and fake (red) samples for different methods at 50K iterations (out of 100K) on CIFAR-10 during training. Gradient norms for GraND/C have a very narrow distribution (spanning less than $\pm 10^{-6}$) around the piecewise Lipschitz constant $K=0.83$. 
Lipschitz Constant Analysis We find that tuning the Lipschitz constant $K$ significantly affects the performance and stability of models when using the Adam optimizer. This is due to the interdependence of $\epsilon_{\text{Adam}}$ in the update with $K$:

$$\delta \theta = -\frac{\langle g \rangle_{\delta_1}}{\sqrt{\langle g^2 \rangle_{\delta_2} + \epsilon_{\text{Adam}}}}. \quad (21)$$

Changing $K$ from its default value of one has an effect of scaling the gradients of the loss function $g$ by $K$. This in turn has an effect of scaling $\epsilon_{\text{Adam}} \rightarrow \epsilon_{\text{Adam}}/K$. Empirically, we find that the Adam update for individual parameters can go $\ll 10^{-7}$ in magnitude on the plateaus of the loss landscape where $\epsilon_{\text{Adam}}$ becomes significant. We find it helps having smaller $K$ generally when training on larger image resolutions which has an effect of increasing $\epsilon_{\text{Adam}}$ from its default value of $1 \times 10^{-8}$. Intuitively, a larger $\epsilon_{\text{Adam}}$ suppresses the Adam update when the loss gradient magnitudes are $\lesssim \epsilon_{\text{Adam}}$. Therefore, $\epsilon_{\text{Adam}}$ determines the extent of noisy Adam updates at plateaus of the loss landscape, and so may need tuning. More details on the choice of $K$ are provided in the supplementary material.

Baselines and Evaluation To directly compare our normalization method GraN with gradient penalty (GP) and spectral normalization (SN), independent of the loss function, we train NSGAN with a gradient penalty $P_1$ loss (NSGAN-GP), and with SNed layers (NSGAN-SN). We also train NSGAN-GP† and NSGAN-SN†, which correspond to models constrained with a tuned Lipschitz constant (instead of 1). WGAN-GP† and SNGAN† are defined analogously.

We quantitatively evaluate the methods by Inception Score (IS) [32], FID [14], and KID [5] with 50K synthetic images randomly sampled from $G$ and 50K real images from the dataset. IS is not used for LSUN and CelebA, as these comprise a single class, for which IS performs poorly [20]. See supplemental material §A for additional details.

Unconditional Image Generation Table 2 presents the comparative results on CIFAR-10, CIFAR-100, and STL-10 image generation. Our methods rank among the top two across every metric (KID, FID, IS) on CIFAR-10 and CIFAR-100. On STL-10, our two methods rank the highest in IS and FID, however, we fall behind on KID by a small margin compared to SNGAN and SNGAN†.

Table 3 summarizes our results on LSUN bedrooms and CelebA image generation. The NSGAN model did not converge (i.e., FID > 70) in two random restarts of training for both LSUN and CelebA. Similarly, WGAN-GP failed to converge on CelebA in two runs. GraND-GAN achieves the best results on CelebA, and the second-best on LSUN bedrooms, falling slightly behind NSGAN-GP†. Among critics, GraNC-GAN performs best by FID as well.

Figure 2 presents a comparison of GraND-GAN with WGAN-GP and SNGAN on CIFAR-10 image generation across various training settings listed in Table 1. For setting

![Figure 4: Boxplots of estimated finite-difference gradient norm with increasing L2 perturbation strengths $\delta$ around the real (blue) and fake samples (red) on CIFAR-10 at 50K iterations (out of 100K) for different methods. Empirically, GraND/C is fairly Lipschitz bounded across polytopes globally, close to the piecewise Lipschitz constant of $K = 0.83$.](image)

$B (n_{\text{dis}} = 1)$, our method retains a respectable FID score (and other metrics) compared to SNGAN and WGAN-GP. For settings C and D with larger learning rates and momentum hyperparameters, the performance of WGAN-GP degrades while our method and SNGAN remain quite robust.

7. Gradient Normalization Empirical Analysis

In this section, we empirically analyze the effect of gradient normalization on (a) the gradient norms and (b) the finite-difference approximation to the gradient norm at increasing levels of perturbation $\delta$, and compare it with spectral normalization and gradient penalty.

Figure 3 shows a boxplot of $\|\nabla_x f(x)\|$ for the baselines, and $\|\nabla_x g(x)\|$ for our methods, on a CIFAR-10 image generation task at 50K iterations (out of 100K), where $f(x)$ is the piecewise linear discriminator (or critic) network and $g(x)$ is its gradient normalized version. The gradient norms $\|\nabla_x f(x)\|$ for NSGAN with an unconstrained discriminator are substantially larger. For WGAN-GP and SNGAN, $\|\nabla_x f(x)\|$ are bounded within a reasonable range. Gradient normalized discriminators and critics with a piecewise Lipschitz constant of $K = 0.83$, have a narrow distribution with a gradient norm that is $\approx K (\pm 10^{-6})$ across samples.

Unlike spectral normalization or gradient penalty, our method does not constrain the discriminator or critic to be globally $K$-Lipschitz. We investigate this on a CIFAR-10 image generation task by first sampling fake data from $G$
Table 2: Unsupervised image generation on CIFAR-10, CIFAR-100, and STL-10. The best and the second best models per evaluation metric and GAN family (i.e., with discriminators or critics) are indicated by bold red and bold blue fonts. † indicates modified baselines with an altered Lipschitz constant $K$. The table is split comparing discriminators (top) and critics (bottom). We write “−” for cases where a model did not achieve a FID $< 70$.

<table>
<thead>
<tr>
<th>Method</th>
<th>IS ↑</th>
<th>FID ↓</th>
<th>KID ↓ (×1000)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CIFAR-10</td>
<td>CIFAR-100</td>
<td>STL-10</td>
</tr>
<tr>
<td>NSGAN</td>
<td>7.655</td>
<td>6.611</td>
<td>7.920</td>
</tr>
<tr>
<td>NSGAN-GP</td>
<td>8.016</td>
<td>−</td>
<td>8.568</td>
</tr>
<tr>
<td>NSGAN-SN</td>
<td>7.792</td>
<td>7.258</td>
<td>8.167</td>
</tr>
<tr>
<td>NSGAN-GP†</td>
<td>8.019</td>
<td>7.892</td>
<td>8.623</td>
</tr>
<tr>
<td>GraN-GAN (Ours)</td>
<td>8.031</td>
<td>8.314</td>
<td>8.743</td>
</tr>
<tr>
<td>WGAN-GP</td>
<td>7.442</td>
<td>7.520</td>
<td>8.492</td>
</tr>
<tr>
<td>SNGAN</td>
<td>8.112</td>
<td>7.778</td>
<td>8.385</td>
</tr>
<tr>
<td>WGAN-GP†</td>
<td>7.344</td>
<td>7.684</td>
<td>8.466</td>
</tr>
<tr>
<td>SNGAN†</td>
<td>7.991</td>
<td>7.959</td>
<td>8.552</td>
</tr>
<tr>
<td>GraN-GAN (Ours)</td>
<td>8.019</td>
<td>7.892</td>
<td>8.623</td>
</tr>
</tbody>
</table>

Table 3: Unsupervised $128 \times 128$ image generation on LSUN-Bedrooms and CelebA. We write “−” to indicate the cases where a model did not achieve a FID $< 70$ in two random training restarts. The best and the second best models per evaluation metric and GAN family (i.e., with discriminators or critics) are indicated by bold red and bold blue fonts. † indicates modified baselines with an altered Lipschitz constant $K$. We split the table into discriminators (top) and critics (bottom), to better highlight the differences per loss function. GraN performs best or second-best across all datasets, losses, and performance metrics; in the case of discriminators, GraN-GAN and NSGAN-GP† (our Lipschitz-tuned GP-based approach) are the top two performers.

<table>
<thead>
<tr>
<th>Method</th>
<th>FID ↓</th>
<th>KID ↓ (×1000)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LSUN</td>
<td>CelebA</td>
</tr>
<tr>
<td>NSGAN</td>
<td>−</td>
<td>−</td>
</tr>
<tr>
<td>NSGAN-GP</td>
<td>−</td>
<td>−</td>
</tr>
<tr>
<td>NSGAN-SN</td>
<td>74.926</td>
<td>14.33</td>
</tr>
<tr>
<td>NSGAN-GP†</td>
<td>10.483</td>
<td>9.385</td>
</tr>
<tr>
<td>NSGAN-SN†</td>
<td>12.635</td>
<td>9.644</td>
</tr>
<tr>
<td>GraN-GAN (Ours)</td>
<td>10.795</td>
<td>9.377</td>
</tr>
<tr>
<td>WGAN-GP</td>
<td>13.562</td>
<td>9.8</td>
</tr>
<tr>
<td>SNGAN</td>
<td>13.237</td>
<td>13.466</td>
</tr>
<tr>
<td>WGAN-GP†</td>
<td>16.884</td>
<td>12.0</td>
</tr>
<tr>
<td>SNGAN†</td>
<td>67.346</td>
<td>15.874</td>
</tr>
<tr>
<td>GraN-GAN (Ours)</td>
<td>12.533</td>
<td>12.000</td>
</tr>
</tbody>
</table>

at a given training iteration and real data from the dataset. We estimate $\Delta_i = \|h(x_i + \delta n_i) - h(x_i)\|/\delta$ for each of the samples, which is the finite difference along a step of magnitude $\delta > 0$ along the local gradient direction $n_i = \nabla h(x_i)/\|\nabla h(x_i)\|$, where $h$ denotes $f$ for baselines and $g$ for our methods, respectively. This provides a probe for the LC (albeit a lower bound) similar in spirit to prior work [43].

Figure 4 shows a boxplot of the resulting $\Delta_i$ for increasing perturbation magnitudes $\delta$. Evidently, gradient normalized discriminators and critics have well-behaved finite-differences, even for fairly large neighborhoods $\delta$, despite being only piecewise Lipshitz in theory. Moreover, the variance of the computed $\Delta_i$’s across $\delta$ for our methods is comparable to WGAN-GP.

8. Discussion

Limitations We observed instabilities when training gradient normalized critics with the Wasserstein loss (Eq. (7)). The hinge loss (Eq. (10)) improved this, but still struggled for larger images; the soft hinge approach was found to work better. However, training GraN with the NS loss (Eq. (2)) was found to be more stable, especially on larger images. Also, on such images, our method periodically diverged late in training, an issue present for the baselines as well.

Future work While GraN does not guarantee a global Lipschitz constraint due to discontinuities, it does enforce constant bounded-norm gradients almost everywhere (and thus piecewise Lipschitz continuity). Moreover, empirically, it is competitive with, or better than, existing baselines. Investigating global versus local Lipschitz continuity, as well as gradient regularization, is thus an enticing future direction.

Conclusion We introduced a novel input-dependent normalization for piecewise linear critics and discriminators. Our method guarantees a bounded input gradient norm almost everywhere and is piecewise $K$-Lipschitz. We empirically showed that our method improves unconditional image generation using GANs across a range of datasets. Finally, though our method does not explicitly impose a global $K$-Lipschitz constraint, empirically, the finite-difference gradient norm is well-behaved in a large local neighbourhood.
References


