Generative Modeling via Feasibility and Iterative Projection

Jacob Abernethy
School of Computer Science
Georgia Institute of Technology
Atlanta, GA 30332
`prof@gatech.edu`

Naveen Kodali
School of Computer Science
Georgia Institute of Technology
Atlanta, GA 30332
`nkodali3@gatech.edu`

Sebastian Pokutta
School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, GA 30332
`sebastian.pokutta@isye.gatech.edu`

Abstract

We consider the problem of modelling a probability distribution from a training set, and we focus on implicit generative models where one aims to directly learn a sampling oracle rather than a density function. We consider the training of such models through a hypothesis testing oracle: given any candidate distribution, we assume we can find potentially violated hypothesis tests, or report that no such hypotheses exist. We show that this framework, which has similarities to Generative Adversarial Networks, leads one to approach generative modelling as a *feasibility problem* rather than an optimization or min-max task, and that this perspective requires different analytical tools and also new estimation methods—in particular, we emphasize the use of *iterative projection* as the core algorithmic tool. We show that this alternative perspective exhibits both statistical benefits but also computational convergence guarantees.

1 Introduction

Density estimation is at the core of data modeling and has a variety of applications like outlier detection, visualization, and statistical testing, and is a key ingredient in addressing inference problems. However, accurate modeling of full probability density function in L1-distance or KL-divergence exhibit statistical challenges in high dimensions. *Implicit* density estimation [22], where one needs only to train a *sampling oracle*, is much more appropriate when our central objective is the ability to draw new samples, and it has found huge application in areas from image processing to speech to text synthesis. Implicit models unfortunately tend to lack the ability to estimate probability densities.

A trend in generative modelling that has taken particular prominence in recent years involves using deep neural networks for estimating distributions. Many of these approaches aim to maximize a likelihood function (belief nets, change-of-variables models, autoregressive methods) or approximations (Boltzmann machines, VAEs). Recently, there has been significant interest in deep implicit models and the most popular algorithmic framework is *Generative Adversarial Networks (GANs)*. For a survey on deep generative models, refer to [9].

The GAN framework [10] is posed as a two player zero-sum game between the *generator* and the *discriminator*. The former is an implicit generative model and it is represented as $G_\phi(z)$ where $z \sim$

Preprint. Work in progress.

$N_d(0, I)$ and ϕ being a parameterization, and the latter is a classifier aimed at distinguishing real samples from those generated by the generator. The latter is represented as $D_\theta(x)$ with parameterization θ. Typically, G and D are chosen to be deep neural networks and they are simultaneously trained using some variant of *Stochastic Gradient Descent (SGD)*. The overall game objective (with log function dropped, see [7, 14]) is -

$$\inf_\phi \sup_\theta \left\{ \mathbb{E}_{z \sim N_d(0,I)}[D_\theta(G_\phi(z))] - \mathbb{E}_{x \sim p}[D_\theta(x)] \right\}, \tag{1}$$

where p is an empirical distribution over a training set of samples.

Despite their apparent empirical success, the GAN framework itself remains somewhat of a mystery, both from a statistical as well as algorithmic perspective. First, given the high-dimensional nature of the output, reasoning about the generalization ability of a GAN output is a significant challenge, which raises many questions on how to regularize and control complexity. The original paper on GANs argues that the minimax objective in (1) is “asymptotically correct” to the extent that in the infinite-data limit the true distribution provides one saddle point solution. But many authors have raised concerns with this argument [1, 2] when data are limited and the output space is high dimensional.

But even putting aside the statistical issues, it is not clear how to find even one saddle point of (1). The popular approach, which is to simultaneously optimize both networks ϕ and θ via SGD, has shown some success in practice with much hand holding, but has frustrated many researchers who observe cycling behavior and other instabilities [8]. Stable convergence and good modeling performance is obtained only in some specialized architectural setups [25] and/or when certain forms of regularization are used for the discriminator [13, 15]; there is no consistent explanation for these phenomena in the literature. As for theoretical results, a recent work [15] gave global convergence guarantees for the simpler convex-concave case and others [24, 21] gave local convergence guarantees but these are based on strong assumptions that typically do not hold in practical settings.

Contributions

1. *Generative Modeling as Feasibility Problem.* In the present paper we aim to provide an alternative view of generative modelling through the lens of *generalized moment matching* when we have oracle access to a class of hypothesis tests. We describe in Section 2 a framework for estimating a distribution via fitting marginals, where the resulting task is the search for a feasible solution in a set of candidate distributions. Each hypothesis test acts as a constraint, and the feasibility tolerance is a parameter tuned for statistical accuracy, in order to avoid the overfitting issues associated with GANs.
2. *Iterative Projections.* In order to solve this feasibility task, we provide in Section 4 a simple yet powerful generic algorithm, based on the method of *iterative projection*, to solve the problem of implicit generative modeling problem that, at least under natural assumptions, guarantee convergence in finitely many steps.
3. *Experiments.* We describe in Section 5 some experiments using our iterative projection ideas and demonstrate superior performance in both the ability to find feasible solutions, as well as the rate of convergence.

2 A Framework for Generative Modelling via Constraint Satisfaction

We begin with several core definitions, setting up the mathematical foundations of the generative modelling problem. We then argue for what we believe should be the key goal of implicit generative modelling. This will lead us to a discussion of uniform deviation, estimation error, and Rademacher complexity.

In our framework, we assume we have some underlying object space $\mathcal{X}$, which one can assume to lie within a compact subset of $\mathbb{R}^d$. Let $\Delta(\mathcal{X})$ denote the space of probability distributions on $\mathcal{X}$. As part of our generative modelling task, we assume we have a set of candidate distributions Q of distributions on $\mathcal{X}$, where $Q \subset \Delta(\mathcal{X})$ is compact. We imagine, a priori, that there is some target distribution $q \in \Delta(\mathcal{X})$ that we want to estimate, yet our only access to q is via a dataset $x_1, \ldots, x_n \sim q$, sampled independently. As such, we define the empirical distribution $\hat{q}_n$ as $\hat{q}_n(A) = \frac{1}{n}\sum_{i=1}^n \mathbf{1}[x_i \in A]$. The goal of generative modelling is to (1) find some $\hat{p}$ which is “close enough” to q, which is (2) learned from $\hat{q}_n$, but (3) is significantly "smoother" than $\hat{q}_n$; we will make these requirements precise in the following.

To get a operational definition of closeness, let us assume that we are given a constrained class of *test functions* $\mathcal{F} := \{f : \mathcal{X} \to [-1, 1]\}$ which help to "distinguish" between distributions. We say that f *ϵ-distinguishes* p from p' if

$$|\mathbb{E}_{x \sim p}[f(x)] - \mathbb{E}_{x' \sim p'}[f(x')]| > \epsilon.$$

Noting that $\mathbb{E}_p[.]$ is a linear operator on $\mathcal{F}$, let us define the operation $f \odot p = p \odot f := \mathbb{E}_{x \sim p}[f(x)]$.

We will assume that $\mathcal{F}$ is bounded and symmetric, i.e., $f \in \mathcal{F} \iff -f \in \mathcal{F}$, and $-1 \leq f(x) \leq 1$ for all $x \in \mathcal{X}, f \in \mathcal{F}$. Indeed we can define a semi-norm on the space of signed measures on $\mathcal{X}$:

$$\|p - p'\|_{\mathcal{F}} := \sup_{f \in \mathcal{F}} \{f \odot (p - p')\}.$$

While it is a slight abuse of terminology, we shall refer to $\| \cdot \|_{\mathcal{F}}$ as a norm throughout, despite that it may not satisfy positive definiteness.

With the above definition in mind, we define the ϵ-ball around a distribution as $B_{\mathcal{F},\epsilon}(p) := \{p' \in \Delta(\mathcal{X}) : \|p' - p\|_{\mathcal{F}} \leq \epsilon\}$.

In this work we assume that the extent to which we can distinguish between distributions p and p' is provided solely by our ability to find a "hypothesis test" $f \in \mathcal{F}$ that can provide a *certificate* of disagreement. To that end, our core algorithmic assumption is that we have access to an oracle $\mathcal{O}_{\mathcal{F}}$ which accomplishes the following: given distributions p, p' and a parameter ϵ as input,

$$\mathcal{O}_{\mathcal{F}}(p, p', \epsilon) \text{ returns } \begin{cases} f \in \mathcal{F} & \text{so that } f \odot (p - p') \geq \epsilon \\ \emptyset & \text{if no such } f \text{ exists} \end{cases}$$

It is worthwhile to mention that this measure-of-closeness is motivated by the literature on hypothesis testing which involves finding *witness functions*, a specific example is the maximum mean discrepancy (MMD) test [11]. In the context of learning densities, such classes of test functions have been used in *tailored* density estimation [26, 27]. More recently, the following papers from GAN literature [7, 18, 17], recent papers on adversarial method of moments [16] and adversarial divergences [14] have used such classes of test functions.

2.1 The Goal of Implicit Generative Modelling

In the statistical literature, much of the focus on estimating probability distributions have used classical notions of distance between measures; the KL-divergence $KL(p||q) := \mathbb{E}_{x \sim p}[\log \frac{q(x)}{p(x)}]$ and the total variational distance $TV(p, q) := \frac{1}{2} \int |p(x) - q(x)| dx$ come to mind. However, in the case of implicit models, where we do not have ready access to probability densities, the former notions of distortion are non-operational. The hypothesis testing framework presented above, on the other hand, dovetails quite well with implicit models. Given two distributions provided to us as sampling oracles G_θ and $G_{\theta'}$, we can quickly estimate $f \odot (G_\theta - G_{\theta'})$ for any $f \in \mathcal{F}$, and thus we can check $\mathcal{F}$-distance quite quickly:

$$\|G_\theta - G_{\theta'}\|_{\mathcal{F}} \leq \epsilon \quad \iff \quad \mathcal{O}_{\mathcal{F}}(G_\theta, G_{\theta'}, \epsilon) \text{ returns } \emptyset.$$

With this in mind, let us ask the following question: *how should we define the goal of implicit generative modelling?* Recall that a generative model takes as input a sample of data points $x_1, \ldots, x_n \sim q$ with empirical distribution given by $\hat{q}_n$ and outputs an estimated model $\hat{p} \in Q$. In this work we argue that the following is the most appropriate definition of success for the modelling task: **the distribution $\hat{p}$ has *succeeded* if the following conditions are met,**

1. $\hat{p}$ is suitably close to $\hat{q}_n$, $\|\hat{p} - \hat{q}_n\|_{\mathcal{F}} \leq \epsilon$, with the choice of ϵ discussed below
2. $\hat{p}$ should be "suitably simple" with respect to some complexity measure $C(\hat{p})$
3. The parameter ϵ should be the smallest possible such that the following constraints are all met with exceedingly high probability
   (a) $B_{\epsilon,\mathcal{F}}(\hat{q}_n) \cap Q \neq \emptyset$,

(b) if we take any other sample of data points $x'_1, \ldots, x'_n \sim q$ with empirical distribution given by $\hat{q}'_n$, then $\|\hat{p} - \hat{q}'_n\|_{\mathcal{F}} \leq \epsilon$,
(c) for the true distribution q we have $\|\hat{q}_n - q\|_{\mathcal{F}} \leq \epsilon$.

Before we continue, it is worthwhile discussing the need for these three conditions on ϵ. The first condition requires simply that the problem is feasible and that we can indeed find some $\hat{p}$ among our space of candidate distributions. The second condition says that we do not want our estimate $\hat{p}$ to be overly sensitive to the particular sample $\hat{q}_n$. The third condition says that we should aim to model our sample $\hat{q}_n$ *only to the extent that* $\hat{q}_n$ actually models the true q. In other words, demanding that our estimate $\hat{p}$ models $\hat{q}_n$ better than $\hat{q}_n$ models q would provide a delicious recipe for overfitting. (It is worth noting that condition (b) is to some extent superfluous. If we know that $\|\hat{p} - \hat{q}_n\|_{\mathcal{F}} \leq \epsilon$ and with high probability $\|\hat{q}_n - q\|_{\mathcal{F}} \leq \epsilon$, then using the triangle inequality we have $\|\hat{p} - \hat{q}'_n\|_{\mathcal{F}} \leq \|\hat{p} - \hat{q}_n\|_{\mathcal{F}} + \|\hat{q}_n - q\|_{\mathcal{F}} + \|\hat{q}'_n - q\|_{\mathcal{F}} \leq 3\epsilon$.)

To summarize what we have thus argued, our claim is that implicit generative modelling from the perspective of hypothesis testing should be viewed squarely as a **constraint satisfaction** or **feasibility problem**. With ϵ suitably chosen, our goal is to find any $\hat{p} \in Q$ such that

$$\hat{p} \in B_{\mathcal{F},\epsilon}(\hat{q}_n) \quad \iff \quad f \odot (\hat{p} - \hat{q}_n) \leq \epsilon \; \forall f \in \mathcal{F}. \tag{2}$$

In other words, each f presents us with a constraint on the estimate $\hat{p}$ and our goal must be to satisfy all such constraints.

Remark: An unavoidable limitation of such learning frameworks based on *hypothesis testing* is that, in the finite sample regime (regardless of the choice of class $\mathcal{F}$), the learned density P_{model} can be quite far from P_{real} in KL-sense. We refer the reader to [11] for an illustrative example which demonstrates that P_{model} can have small support.

2.2 The choice of ϵ: theoretically and empirically

In the previous subsection we observed that the choice of ϵ is key for obtaining the correct approximation error we hope to achieve. In short, we want to select ϵ to be suitably small to ensure that we are correctly approximating q, but not so small that we are simply fitting the empirical distribution $\hat{q}_n$. We will now formalize how to select the tolerance parameter and we will consider both a theoretically motivated choice, as well as an empirical solution that is more suited for practical computations.

Let $\delta > 0$ be some tolerated error probability. We define the *worst case error tolerance* $\epsilon^*_{\mathcal{F},n}$ as the infimum over $\epsilon > 0$ that satisfies

$$\text{for every } p \in \Delta(\mathcal{X}) : \qquad \Pr_{\hat{p}_n \sim p}\left[\|p - \hat{p}_n\|_{\mathcal{F}} \leq \epsilon\right] \geq 1 - \delta,$$

where $\hat{p}_n \sim p$ means that $\hat{p}_n$ is the empirical distribution of an IID sample drawn from p.

This definition of an "optimal" ϵ is perhaps overly-pessimistic, as it demands that the threshold should hold for *any* distribution $p \in \Delta(\mathcal{X})$. On the other hand, there is a well-developed toolkit for tightly estimating what this $\epsilon^*_{\mathcal{F},n}$ will be. This is the subject of *uniform deviation bounds*, which aim to control the size of the largest deviation of an empirical average $\frac{1}{n}\sum_{i=1}^n f(x_i)$, on a sample drawn from p, from the true mean $\mathbb{E}_{x\sim p}[f(x)]$. For example, a classical result going back to Vapnik and Chervonenkis [23] states that, for a binary valued class $\mathcal{F}$, we have that for some constant $c > 0$ with probability at least $1 - \delta$ it holds $\|p - \hat{p}_n\|_{\mathcal{F}} \leq c\sqrt{\frac{\text{VCdim}(\mathcal{F}) \log n}{n}} + \sqrt{\frac{\log 1/\delta}{2n}}$, where $\text{VCdim}(\mathcal{F})$ is the *Vapnik-Chervonenkis dimension* of the class of hypotheses $\mathcal{F}$. These bounds were exploited by [6] in the regularized version of maximum entropy density estimation problem.

We can use a more precise estimate for ϵ that is tailored for the distribution q and uses the actual samples $x_1, \ldots, x_n$. We define the *empirical Rademacher complexity* [3] of $\mathcal{F}$ given $x_1, \ldots, x_n$ as

$$\widehat{\text{Rad}}_n(\mathcal{F}; x_1, \ldots, x_n) := \mathop{\mathbb{E}}_{\sigma_1,\ldots\sigma_n \sim\{-1,1\}}\left[\sup_{f\in\mathcal{F}} \tfrac{1}{n}\textstyle\sum_{i=1}^n \sigma_i f(x_i)\right],$$

where the σ_i's are drawn IID and uniformly at random from $\{-1, 1\}$. One is then able to show [23] that with probability at least $1 - \delta$,

$$\|p - \hat{p}_n\|_{\mathcal{F}} \leq \widehat{\text{Rad}}_n(\mathcal{F}; x_{1:n}) + 3\sqrt{\tfrac{\log 2/\delta}{2n}}.$$

And as such, put simply, the right hand side makes a good choice for ϵ.

In practice, of course, the above estimates may be hard to obtain, since computing the VC dimension and/or the (empirical) Rademacher complexity can be non-trivial. But one can obtain a reasonable empirical estimate using a bootstrapping-like technique: for a small positive integer k, partition the dataset into k disjoint subsets with empirical distributions $\hat{q}^{(1)}, \ldots, \hat{q}^{(k)}$, and then choose $\hat{\epsilon} := \max_{i \neq j} \|\hat{q}^{(i)} - \hat{q}^{(j)}\|_{\mathcal{F}}$. This is indeed how we tuned ϵ in our experiments.

Now, given this formulation, we will address the algorithmic question of how we might be able to satisfy a potentially infinite set of constraints in Section 4.

3 Generative Adversarial Networks and their Challenges

Let us return our attention to the GAN objective discussed in the introduction. A careful reader might observe that the minimax problem in (1) has some resemblance to the framework we laid out in section 2. Indeed, if we replace $D_\theta(\cdot)$ with $f(\cdot)$, and simply refer to the distribution of the output $G_\phi(z)$ as $\hat{p}$, then we obtain

$$\inf_{\hat{p} \in \{G_\phi : \phi \in \Phi\}} \sup_{f \in \mathcal{F}} \{f \odot (\hat{q}_n - \hat{p})\} \quad = \quad \inf_{\hat{p} \in \{G_\phi : \phi \in \Phi\}} \|\hat{q}_n - \hat{p}\|_{\mathcal{F}}. \tag{3}$$

The right hand side of the above expression is very similar to the constraint satisfaction task (2) we formulated before, but it is different in several critical ways. To summarize, in our framework we want to solve

$$\text{Find } \hat{p} \text{ satisfying} \quad \|\hat{p} - \hat{q}_n\|_{\mathcal{F}} \leq \epsilon \qquad \text{where} \begin{cases} \text{(a) } \epsilon \text{ chosen precisely to avoid overfitting AND} \\ \text{(b) selection of } \hat{p} \text{ biased towards low complexity.} \end{cases}$$

This brings us to what we call the **GAN statistical challenge**. Notice that our formulation specifies that we should seek a $\hat{p}$ which is *suitably close* to $\hat{q}_n$, but only to an extent that is statistically reasonable. On the other hand the GAN objective demands similarity to $\hat{q}_n$ as the primary object to be minimized, despite that this could lead to severe overfitting. The original argument for the GAN objective [10] relied on the infinite-data regime whereas, following our discussion in Section 2, in the finite data regime one has to choose the optimization tolerance ϵ carefully according to statistical properties of the discriminator class and the dataset size, and anything beyond this would certainly lead to an overfit model.

We face another purely algorithmic challenge and we will refer to this as the **GAN optimization challenge**. For this consider the minimax problem in (1), and let $g(\phi, \theta)$ refer to the objective function. The standard optimization strategy is to perform gradient *descent* on the ϕ parameters, while simultaneously performing gradient *ascent* on the θ parameters. We face the natural question: how do we measure progress at some iteration t for the parameters ϕ_t, θ_t? In contrast to usual optimization that one encounters in supervised learning settings, where we can simply monitor the decrease in average training loss, for the GAN objective it is unclear what the value $g(\phi_t, \theta_t)$ actually signifies. While it is known that some *saddle point* ϕ^*, θ^* exists (i.e., where ϕ^*, θ^* are a local minimum and maximum, respectively), simultaneous gradient ascent/descent is *not guaranteed* to find such a pair. We stress that this is not merely a pathological challenge: it is straightforward to find simple examples (see Section 5.1) where cycling is not just possible but *guaranteed* and cycling is also regularly observed in practice.

One possible solution to the GAN optimization issue, proposed by [15, 12], is to use regret minimization tools to solve for a saddle point of the game. While a nice idea in theory, it presents at least two significant challenges. First, this approach requires that one stores all parameter iterates throughout the optimization process, which can be prohibitively expensive. Second, it requires taking cumulative averages of stored parameters, which only really makes sense in the context of convex loss functions.

In the following section we will propose a framework to resolve both the statistical as well as algorithmic challenges for training implicit generative models.

4 Iterative Projection Algorithms for Feasibility Problems

We will now explore the framework of iterative projection to solve feasibility problems, and in particular to obtain precise guarantees on convergence. First, we present an easy example, the

Winnow algorithm, that can solve linear feasibility problems in the space of distributions, where the rate depends on the inverse square of the size of the margin of feasibility. This exhibits a somewhat striking fact (given some fixed amount of data): the greater the complexity in the hypothesis class $\mathcal{F}$, the larger the margin ϵ, which actually leads to a *faster rate* of convergence in terms of the number of required iterations.

4.1 The Winnow Algorithm

The Winnow Algorithm can be interpreted as solving a generative modeling problem which motivates the later discussion about iterative projections. As discussed earlier, our goal is to solve the linear feasibility problem (2). However the constraint $f \odot (\hat{p} - \hat{q}_n) \leq \epsilon$ is unfortunately not homogeneous, so we can make the following modification to our function space $\mathcal{F}$: let $\tilde{\mathcal{F}} := \{\tilde{f}(x) = f(x) + \epsilon - f \odot \hat{q}_n : f \in \mathcal{F}\}$. Now we can define an equivalent set of homogeneous constraints,

$$\hat{p} \in B_{\mathcal{F},\epsilon} \quad \Longleftrightarrow \tilde{f} \odot \hat{p} \geq 0 \quad \forall \tilde{f} \in \tilde{\mathcal{F}}.$$

For such problems, a simple algorithm, the so-called Winnow algorithm [19] is effective when there is a feasible solution with large margin.

Algorithm 1 Winnow Algorithm

```
Input: tolerance $\epsilon > 0$, oracle $\mathcal{O}_{\mathcal{F}}$, init distribution $p_0$ = Unif, training set $\hat{q}_n$, learning rate $\eta > 0$
for $t = 1, 2, \ldots, T$ do
    if $\mathcal{O}_{\mathcal{F}}(p_{t-1}, \hat{q}_n, \epsilon) \to f_t \equiv \emptyset$ then
        Our estimate satisfies $\|p_{t-1} - \hat{q}_n\|_{\mathcal{F}} \leq \epsilon$, hence Return $p_{t-1}$
    else
        Update $p_t$:    $p_t(x) = \frac{p_{t-1}(x)\exp(\eta \tilde{f}_t(x))}{Z_{t-1}}$    where $Z_{t-1}$ is the normalization factor.
    end if
end for
```

Lemma 1. Algorithm 1 terminates in no more than $O\left(\frac{\log \frac{2}{\epsilon}}{\epsilon^2}\right)$ iterations and the returned distribution p_T is in the desired set $B_{\mathcal{F},\epsilon}(\hat{q}_n)$.

Proof. (sketch) We can follow the standard analysis of Winnow, but we need to compare our sequence of distributions to an "optimal" point $\tilde{q}$. For analysis, let us choose $\tilde{q} = (1 - \epsilon/2)\hat{q}_n + (\epsilon/2)$Unif, where Unif is the uniform distribution. This choice ensures two properties: (a) $\tilde{q}$ is at least $\epsilon/2$-feasible for every constraint $\tilde{f} \in \tilde{\mathcal{F}}$, and (b) we can bound the KL divergence as $KL(\tilde{q}||p_0) \leq \log \frac{2}{\epsilon}$, since $\tilde{q}$ has a "large enough" mixture with the uniform distribution. The standard analysis of Winnow guarantees that, as long as $\tilde{q}$ satisfies every constraint with margin at least $\gamma > 0$ then the number of updates required to find a feasible $\hat{p}$ is no more than $\frac{1}{\gamma^2} KL(\tilde{q}||p_0)$. □

Observe that the updates of the Winnow algorithm can be obtained by solving the following minimization problem in each round, which is in spirit a mirror descent-like step: $p_t = \min_p D_{KL}(p, p_{t-1}) - \eta \cdot \tilde{f} \odot p$.

4.2 Projection methods

In this section we will now consider more general methods to solve feasibility problems via iterative projection. This generalizes the Winnow approach, which is well-suited for linear constraints, to more general convex constraints. At a high level, projection methods are a class of algorithms that are used to solve convex feasibility problems, where one hopes to obtain a point in the intersection of a family of closed convex sets in a Hilbert space. They are especially effective in solving problems where individual sets are specified by linear inequalities [5] (which happens to the case for our problem (2)).

Definition 4.1. Let $S \subseteq \mathcal{X}$ be a closed convex set, let $x \in \mathcal{X}$, and let $\|.\|$ be a norm. Then the *projection* of x onto S with respect to $\|.\|$ is defined as $\hat{x} := \text{argmin}_{s \in S} \|x - s\|$.

Let the closed half-spaces induced by our constraints be labeled as $S_1, ..., S_N$ with non-empty intersection $\emptyset \neq S := \bigcap_{i \in [N]} S_i$. Our goal is find some point $q \in S$. Iterative projection algorithms maintain a sequence of points $q_1, \ldots, q_T \subseteq \mathcal{X}$. Given the current iterate q_t, the next iterate q_{t+1} is obtained by solving

$$q_{t+1} \leftarrow P_i q_t,$$

where P_i is the projection onto S_i and the set to project on is decided by what is termed as a *control scheme*. Such individual projections in our case are easy to perform. Given a halfspace S_i specified by a linear inequality $\langle a_i, q \rangle \leq b_i$ with $a_i \in \mathbb{R}^{|X|} \setminus \{0\}$ and $b_i \in \mathbb{R}$, we can compute the projection for any $q_t \in \mathbb{R}^{|X|}$ in closed form

$$P_i q_t = q_t - \tfrac{(\langle a_i, q_t \rangle - b_i)^+}{||a_i||^2} a_i.$$

The following very useful property called *linear regularity* holds for collections of half-spaces.

Definition 4.2. Let $d(.,.)$ be a distance. An N-tuple of closed convex sets $(C_1, ..., C_N)$ is *linearly regular* if there exists $\kappa \geq 1$ such that for all $q \in \mathbb{R}^{|X|}$ we have $d(q, C) \leq \kappa \max_{j \in \{1,...,N\}} d(q, C_j)$, where $C = \bigcap_{i \in [N]} C_i$.

Linear regularity ensures that a point p that is close to all individual sets S_i is also close to their intersection S and it guarantees that projection algorithms converge linearly fast to a point in S [4]. Further, one can use a simple control scheme $q_{t+1} \leftarrow P_{i^*} q_t$, where $i^* := \max_{j \in \{1,...,N\}} d(q_t, C_j)$ to achieve this rate. As a consequence our iterative projection algorithm only requires access to the most-violated constraint at each step. Empirically, access to "representative" constraints in each iteration typically suffices for fast convergence.

We are ready to provide a template for our *Iterative Projection* in Algorithm 2 for our feasibility problem (2).

Algorithm 2 Iterative Projection Template

```
Input: tolerance $\epsilon > 0$, oracle $\mathcal{O}_{\mathcal{F}}$, initial distribution $p_0 \in Q$, distance function $d(\cdot, \cdot)$
for $t = 1, 2, \ldots, T$ do
    if $\mathcal{O}_{\mathcal{F}}(p_{t-1}, \hat{q}_n, \epsilon) \to f_t \equiv \emptyset$ then
        Our estimate satisfies $\|p_{t-1} - \hat{q}_n\|_{\mathcal{F}} \leq \epsilon$, hence Return $p_{t-1}$
    else
        Set $\hat{p}_t$ minimizing $\min_{p \in Q} d(p, \hat{p}_{t-1})$   such that   $(p - \hat{q}_n) \circ f_t \leq \epsilon$
    end if
end for
```

Lemma 2. If our initial solution p_0 satisfies $d(p_0, S) \leq D$ for some D, then Algorithm 2 convergences to a point $s \in B_{\mathcal{F},\epsilon}(\hat{q}_n)$ and in particular $d(s, S) \leq \varepsilon$ within $O\left(\kappa_{\mathcal{F}}^2 \log \frac{D}{\epsilon}\right)$ iterations, where $\kappa_{\mathcal{F}} \geq 1$ is the parameter from the linear regularity condition.

We will omit a proof of the above lemma for space, but it is reasonably standard and can be found in [4] (see Theorem 5.8 in the reference). It is important to note that, while the result is stated in terms of an oracle that finds the most-violated constraint, one can relax this condition and prove a similar result for an oracle that finds an approximate worst constraint satisfying linear regularity like condition with some $\rho_{\mathcal{F}} > \kappa_{\mathcal{F}}$ (convergence rate will now depend on ρ instead). Finally, we would like to mention that it is possible obtain simple reformulations of (2) that ensure that the intermediate iterates of IP lie on the probability simplex.

4.3 Updates in parameter space using implicit generative models

In our analysis above, we framed the problem as projections accomplished in *probability space*, in that we are operating on the densities $\hat{p}$ themselves. While this provides us with very nice theoretical results on convergence, particularly because the constraints $f \in \mathcal{F}$ are indeed *linear* on the space of densities, we are in practice limited to optimization in parameter space. This has the limitation that constraints f are potentially non-convex in the space of parameters, but what we will show empirically is that this does not appear to dramatically affect convergence rates.

The iterative projection scheme discussed above requires one critical choice: the distance function $d(\cdot,\cdot)$. We will now consider three separate such distance (or distance-like) metrics on the space of parameters. We are going to focus on the class of deep implicit generative models with ReLU activation layers. These are parameterized as follows: $G_{\theta=(\theta_1,\ldots,\theta_K)} = \tau(\theta_K(\tau(\ldots\tau(\theta_1 z)\ldots)))$, where τ is the ReLU activation function (mapping each coordinate of the input $x \mapsto \max(x,0)$) and each θ_i is a matrix with appropriately matching shapes. Let us now consider

1. **KL-divergence.** An idealized choice for a distance measure is $d(p_\theta, p_{\theta'}) = KL(p_\theta || p_{\theta'})$. Projection with this metric would effectively give us the Winnow algorithm discussed above. Of course computing entropy, let alone KL divergence, is known to be a challenge for such implicit models.
2. **Pseudo-KL**. The KL divergence can be represented as a *Bregman divergence* with respect to the entropy function, i.e., $KL(p||q) = D_H(p,q) := H(p) - H(q) - \nabla H(q)\cdot(p-q)$, where H is the Shannon entropy function. An alternative is something we might call the *pseudo-entropy* of p_θ:

   $$\tilde{H}(\theta_1,\ldots,\theta_K) := \sum_{i=1}^{K} \log|\det(\theta_i^\top \theta_i)|,$$

   (one should use $\theta_i\theta_i^\top$ instead depending on whether there are more rows or columns). With this in mind, the pseudo-KL distance is the Bregman divergence of the parameters for $\tilde{H}$, i.e., $d(p_\theta, p_{\theta'}) := \widetilde{KL}(\theta,\theta') = \tilde{H}(\theta) - \tilde{H}(\theta') - \nabla\tilde{H}(\theta')\cdot(\theta-\theta')$. This is convenient as the derivative is easy to calculate: $\nabla \log\det M = M^{-1}$ for a symmetric matrix M.

   We must emphasize that this is a well-motivated notion of entropy on the generator's parameters. If Z is any absolutely continuous random variable, then $H(MZ) = H(Z) + \log|\det M|$; that is, the matrix multiplication increases/decreases entropy by the log of the determinant of the transformation. The operation $\tau(\cdot)$ for the ReLu only *decreases* entropy, since it "zeros out" coordinates. This argument implies that $\tilde{H}(\theta)$ is a natural upper bound on $H(p_\theta)$.
3. **Average output distance.** Another natural distance between distributions is the following:

   $$d(p_\theta, p_{\theta'}) = \mathbb{E}_{z\sim N(0,I)}[||G_\theta(z) - G_{\theta'}(z)||],$$

   for some norm $||\cdot||$ on $\mathcal{X}$. Simply put, this checks how far the parameters change the sampling oracles output, and it can be estimated via sampling. While this is certainly a heuristic (inspired from Euclidean projection in the probability space), it exhibits reasonably good empirical performance.

5 Experiments

In this section, we describe two experiments to demonstrate the differences between our iterative projection framework and the GAN framework. The first one is an analytic experiment while the second one is a computational experiment.

5.1 Analytical Experiment with Dirac distributions

Let us consider a simple analytical example to demonstrate how our new framework overcomes the shortcomings of the GAN framework. We use the example from [20] (Definition 2.1), where the learning of univariate Dirac distributions is considered. The true distribution is given by δ_0, the generator class is δ_ϕ and the discriminator class is all the linear separators of the form $\theta\cdot x$. Here, the GAN game simplifies to $\min_\phi \max_\theta \theta\cdot\phi$. There is a unique equilibrium $\phi = \theta = 0$ in this game. However, if both the players use simultaneous gradient descent/ascent, it can be easily shown that the players either cycle indefinitely or diverge from the solution. We refer the reader to [20, 8] for details.

Next, let us analyze how our iterative projection framework fares in this example. The setup can be transformed into that of a feasibility problem as

$$\text{Find } \phi \quad \text{s.t} \quad \theta\cdot\phi \le 0 \quad \forall\, \theta$$

Here, ϵ can be set to zero as we have full information about the real distribution. We start with some initial generator model δ_{ϕ_0} (assume ϕ_0 > 0 w.l.o.g.) and call the separation oracle to obtain a violated constraint, i.e., some $\theta_1 \cdot x$ such that $\theta_1 \cdot \delta_{\phi_0} > 0$. This is easy to find as any $\theta > 0$ will work. Now, we claim that if the generator performs an Euclidean projection onto the constraint $\theta_1 \cdot \phi \leq 0$, we will reach the desired of $\phi^* = 0$ in each step. This is because the projection step involves solving the optimization problem $\min_\phi ||\delta_\phi - \delta_{\phi_0}||$ s.t $\theta_1 \cdot \phi \leq 0$, which has a unique solution $\phi^* = 0$. No further updates will be made to the generator since all the constraints are now satisfied.

This example while being simplistic is albeit prototypical of the settings in which GANs are used and such cycling behavior indeed carries over to practical settings. In contrast, as we demonstrated, our iterative projection framework would be more effective and stable.

5.2 Experiments with Toy Datasets

We setup the following experiment using IID samples from the toy datasets: a mixture of 8 gaussians and a swissroll. The architecture for both the generator and the discriminator is chosen to be a standard four layer MLP. For training, in each round, D uses 1 SGD step with a small gradient penalty [15, 13] added (this seems to help in satisfying the condition 3a from section 2.1). We allow the training to proceed for some fixed number of rounds. The algorithms we compare differ in how we update G in each round -

1. In the typical GAN setup, G uses 1 SGD step.

2. In iterative projection, G projects onto the discriminator constraint if it is found to be violating. We experiment with average output distance and the pseudo-KL discussed in Section 4.3 and run SGD for some larger number of rounds.

3. To establish that it is crucial for G to perform projections, we also run an experiment using the GAN algorithm but with G trained to "optimality" at each round.

We show the final model distribution learned in each case in Figure 1.

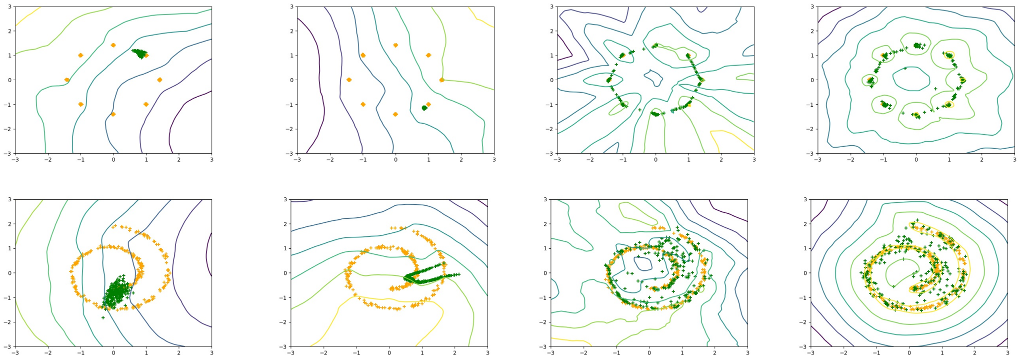

Figure 1: We show the final model distribution learned using four different methods: GAN, GAN with optimal generator, IP with average output distance and IP with pseudo-KL (from left to right) and for two different datasets (top is GMM and bottom is swissroll). In the figures, orange is used to denote real samples, green for generated samples, and contour plots show the final discriminator function.

References

[1] Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (gans). *arXiv preprint arXiv:1703.00573*, 2017.

[2] Sanjeev Arora and Yi Zhang. Do gans actually learn the distribution? an empirical study. *arXiv preprint arXiv:1706.08224*, 2017.

[3] Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

[4] Heinz H Bauschke and Jonathan M Borwein. On projection algorithms for solving convex feasibility problems. *SIAM review*, 38(3):367–426, 1996.

[5] Yair Censor, Wei Chen, Patrick L Combettes, Ran Davidi, and Gabor T Herman. On the effectiveness of projection methods for convex feasibility problems with linear inequality constraints. *Computational Optimization and Applications*, 51(3):1065–1088, 2012.

[6] Miroslav Dudik, Steven J Phillips, and Robert E Schapire. Performance guarantees for regularized maximum entropy density estimation. In *International Conference on Computational Learning Theory*, pages 472–486. Springer, 2004.

[7] Gintare Karolina Dziugaite, Daniel M Roy, and Zoubin Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. *arXiv preprint arXiv:1505.03906*, 2015.

[8] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[9] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[11] Arthur Gretton, Karsten Borgwardt, Malte J Rasch, Bernhard Scholkopf, and Alexander J Smola. A kernel method for the two-sample problem. *arXiv preprint arXiv:0805.2368*, 2008.

[12] Paulina Grnarova, Kfir Y Levy, Aurelien Lucchi, Thomas Hofmann, and Andreas Krause. An online learning approach to generative adversarial networks. *arXiv preprint arXiv:1706.03269*, 2017.

[13] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5769–5779, 2017.

[14] Gabriel Huang, Gauthier Gidel, Hugo Berard, Ahmed Touati, and Simon Lacoste-Julien. Adversarial divergences are good task losses for generative modeling. *arXiv preprint arXiv:1708.02511*, 2017.

[15] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. How to train your dragan. *arXiv preprint arXiv:1705.07215*, 2017.

[16] Greg Lewis and Vasilis Syrgkanis. Adversarial generalized method of moments. *arXiv preprint arXiv:1803.07164*, 2018.

[17] Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabás Póczos. Mmd gan: Towards deeper understanding of moment matching network. In *Advances in Neural Information Processing Systems*, pages 2200–2210, 2017.

[18] Yujia Li, Kevin Swersky, and Rich Zemel. Generative moment matching networks. In *International Conference on Machine Learning*, pages 1718–1727, 2015.

[19] Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine learning*, 2(4):285–318, 1988.

[20] Lars Mescheder. On the convergence properties of gan training. *arXiv preprint arXiv:1801.04406*, 2018.

[21] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. The numerics of gans. In *Advances in Neural Information Processing Systems*, pages 1823–1833, 2017.

[22] Shakir Mohamed and Balaji Lakshminarayanan. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*, 2016.

[23] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2012.

[24] Vaishnavh Nagarajan and J Zico Kolter. Gradient descent gan optimization is locally stable. In *Advances in Neural Information Processing Systems*, pages 5591–5600, 2017.

[25] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[26] John Shawe-Taylor and Alex Dolia. A framework for probability density estimation. In *Artificial Intelligence and Statistics*, pages 468–475, 2007.

[27] Le Song, Xinhua Zhang, Alex Smola, Arthur Gretton, and Bernhard Schölkopf. Tailoring density estimation via reproducing kernel moment matching. In *Proceedings of the 25th international conference on Machine learning*, pages 992–999. ACM, 2008.