Convergence and Optimality Analysis of Low-Dimensional Generative Adversarial Networks Using Error Function Integrals

GRAHAM W. PULFORD, (Senior Member, IEEE), AND KIRILL KONDRASHOV, (Member, IEEE)

1 Consultant, BandGapAI, 31170 Toulouse, France
2 GN A/S, 5613 Eindhoven, The Netherlands

Corresponding author: Graham W. Pulford (gpulford@ieee.org)

The work of Kirill Kondrashov was supported by GN A/S.

ABSTRACT Due to their success at synthesising highly realistic images, many claims have been made about optimality and convergence in generative adversarial networks (GANs). But what of vanishing gradients, saturation, and other numerical problems noted by AI practitioners? Attempts to explain these phenomena have so far been based on purely empirical studies or differential equations, valid only in the limit. We take a fresh look at these questions using explicit, low-dimensional models. We revisit the well known optimal discriminator result and, by construction of a counterexample, show that it is not valid in the case of practical interest: when the dimension of the latent variable is less than that of the data: \( \text{dim}(z) < \text{dim}(x) \). To examine convergence issues, we consider a 1-D least squares (LS) GAN with exponentially distributed data, a Rayleigh distributed latent variable, a square law generator and a discriminator of the form \( D(x) = (1 + \text{erf}(x))/2 \) where \( \text{erf} \) is the error function. We obtain explicit representations of the cost (or loss) function and its derivatives. The representation is exact down to the evaluation of a well-behaved 1-D integral. We present analytical numerical examples of 2D and 4D parameter trajectories for gradient-based minimax optimisation. Although the cost function has no saddle points, it generally has a minimum, maximum and plateau areas. The gradient algorithms typically converge to a plateau, where the gradients vanish and the cost function saturates. This is an undesirable setting with no implications of optimality for either the generator or discriminator. The analytical method is compared with stochastic gradient optimisation and proven to be a very accurate predictor of the latter’s performance. The quasi-deterministic framework we develop is a powerful analytical tool for understanding convergence behaviour of low-dimensional GANs based on least-squares cost criteria.

INDEX TERMS Alternating optimization, change of variable, convergence, cost surface, counterexample, error function integral, GAN, generative adversarial network, gradient descent, least squares GAN, low-dimensional model, optimality, optimal discriminator, optimal generator, performance analysis.

I. INTRODUCTION

Generative adversarial networks (GANs) employ a minimax or game-theoretic framework to derive a mapping from a compressed space of “latent variables” to the space of the 2-D image data. The unsupervised learning problem is configured so that the synthetic images generated by the GAN are as close as possible to the original data, as measured by the ability of a discriminator to distinguish between real and synthetic images. The formulation is actually variational, i.e., over a space of functions, rather than parametric. Its practical implementation uses neural networks to represent the discriminator and generator functions and training is carried out over the set of parameters via alternating stochastic gradient descent (SGD). The GAN framework has met with widespread acceptance in the artificial intelligence (AI) community since its appearance in 2014 [1]. When coupled with the representational capabilities of deep
convolutional neural networks (CNNs) [2], GANs provides a powerful basis for inferential modelling of 2-D image data that is of great practical value in many application areas [3].

The spectacular uptake of the adversarial framework in deep learning is evidenced by the appearance of numerous recent surveys, such as [4], [5], which have lent credibility to the GAN theory developed in [1]. In particular, the derivation of the “optimal discriminator” (Proposition 1 in [1]), with mapping $D_G(x)$ for a given generator function $G(x)$, given by

$$ D_G(x) = \frac{p_d(x)}{p_d(x) + p_g(x)} $$

where $p_d(x)$ is the data PDF and $p_g(x)$ is the generator output PDF, has appeared in a large number of publication on GANs. The same type of derivation appears in a slightly modified form, switching $p_g$ and $p_d$ in (1), in the case of the least squares GAN (LSGAN) in [6].

Numerous researchers, on the other hand, have pointed to troubling numerical problems arising in the training of GANs. Among the better known ones are: convergence failure, where the SGD algorithm stalls due to near zero gradient [7], and mode collapse [8]–[10] where only a subset of a multi-modal data distribution is captured during generator training [11], including the case of collapse to a single point [2]. Cao et al. [9] also refer to problems with vanishing gradients and “poor diversity.” A number of authors have conducted simulations indicative of discrepancies between the observed training behaviour of GANs and the claimed theoretical results. Arjovsky et al. [12] point to the “satisfaction” of the Jensen-Shannon distance, which shows poor correlation with the generated image quality. The presence of serious numerical instabilities has led to a host of new formulations of GAN-like cost functions attempting to improve on the original GAN framework (see, for example, [11], [13]), with [14] listing more than 70 types. Very elaborate schemes have been proposed, including evolutionary variants using multiple models for both generator and discriminator [15], [16].

There have been many efforts to understand the learning dynamics for gradient descent schemes in various types of feedforward multilayer perceptron (MLP) neural networks [17]–[20], which frequently use reduced-dimension models or simplifications of the network structure (e.g., a small number of neurons) to facilitate analysis and visualisation. Amari et al. [21] investigated singular regions in the parameter space of a three-layer MLP network and linked the presence of saddle points having near-zero eigenvalues to plateaux phenomena. In such a situation, parameter adaptation can stall for a long time until the edge of the plateau is reached, where gradient descent recommences. An analysis of the dynamics of these MLP networks is practicable by transforming the parameters to a space that separates the fast and slow dynamical components (see [22], ch. 12.2). Amari and his co-workers as well as Tsutsui [23] apply centre manifold theory to analyse low-dimensional dynamical systems models of 3-layer MLPs in continuous time.

The situation for GANs is somewhat different due to the form of the cost function, which involves expectations of the discriminator and generator functions, where the latter depend on a set of parameters that define the network. Although the generator prior is known, the data PDF is not; so there is a fundamental uncertainty in the cost function, which cannot generally be expressed in a closed form. For this reason, to date, researchers seeking to understand and improve GANs have generally relied on purely empirical (simulation-based) evidence, or have taken as the starting point for their analyses the results in [1] concerning the optimal discriminator, which has been used to fix the form of the cost function. A case in point is Arjovsky and Bottou [24], who seek to explain the vanishing gradient problem that can occur when training the generator, leading to the introduction of the Wasserstein metric, which they advocate for their modified WGAN. A recent work [25] proposes a variant called a message importance measure (MIM) GAN, which uses exponentials rather than logarithms in the cost function to more accurately reproduce the tails of the data distribution. Both the WGAN and MIM-based GAN make use of an optimal discriminator assumption in their analyses.

A slightly different approach appears in a series of papers [26]–[28] that seek to model GAN training dynamics by an associated system of ordinary differential equations (ODEs). The ODE system is derived from the alternating gradient optimisation equations for the GAN in the limit of zero step size. The validity of such an approximation is questionable since GANs use a non-zero step size and the original framework is a stochastic approximation to a gradient algorithm, although the latter point is partly addressed in [27]. As mooted before, a more serious problem is the lack of an explicit cost function. This means that the associated system of ODEs can only be derived when the data PDF is known. Subsequent generic arguments for stability or convergence are somewhat circular due to the fact that the system of ODEs has no explicit form.

In the explicit examples provided so far, Mescheder et al. [28] construct a 1-D “Dirac GAN” in which the generator PDF is a Dirac delta function with a linear discriminator $D(x) = ax$. Nagarajan and Kolter [26] provide a 1-D example based on a “logistic” discriminator $1/(1 + \exp(-wx^2))$, a linear generator $G(z) = az$ with both data and latent variable PDFs uniform on $[-1, 1]$. As both systems have only two parameters (one for the generator and one for the discriminator), their orbits can be visualised as phase plane portraits [29]. These particular examples are perhaps too simple to shed light on the training dynamics of a GAN when the step size is finite, and both assume that a Nash equilibrium exists. Other worrying assumptions include the “perfect generator” assumption, where $p_g \equiv p_d$, which assumes convergence to the desired solution. A more recent effort in [30], which does not suffer the aforementioned drawbacks, modifies the cost functional in the discriminator...
optimisation for compatibility with logistic regression, and replaces the generator optimisation with a composite functional gradient (CFG) scheme. Under certain assumptions the resulting CFG GAN can be shown to minimise the KL divergence between the real and generated data distributions. According to [30], the standard GAN generator update is a coarse approximation to the incremental version of CFG.

Difficulties arising in GAN training have led researchers to question the existence of Nash equilibria in the context of the game-theoretic framework. Farnia and Ozdaglar [31] present convincing empirical evidence to show that two variants of the Wasserstein GAN do not converge to a Nash equilibrium in a number of experimental settings. The latter work utilises a modified GAN cost functional that seeks a different type of equilibrium called a proximal equilibrium. Sidheekh et al. [32] build on this idea through the introduction of a duality gap on the proximal equilibrium, which seems to be a more reliable indicator of GAN convergence under alternating gradient descent optimization. Further research that is relevant in the present context includes [33], [34] and [35]. In [33], a convergence result is established for generators that are 1-layer neural networks with the same number of inputs as outputs. Sun et al. [34] seek to understand the “loss landscape” of the original GAN for the discriminator optimisation. They show that the empirical discriminator loss function has multiple sub-optimal local minima, each corresponding to a mode-collapse situation. Manisha et al. [35] consider subjective (visual) performance of various GANs when the dimension of the latent (noise) variable is varied.

In this article, we seek to clarify the theoretical basis for GANs, concentrating on questions of optimality and convergence. We take a fresh look at the theory presented to justify the original framework. We contend that there is a link between the numerical instabilities already noted by other researchers and the theoretical inconsistencies in GANs that we expose in this article. Our approach differs significantly from previous works based either on empirical evidence obtained during GAN training or resorting to continuous-time ODEs. The latter are not necessarily representative of actual GAN behaviour while the former expose various numerical problems but do not provide a means for their elucidation due to the sheer complexity of the training dynamics of neural networks. Equally, our approach contrasts with that of “GAN Lab” [36], which is a web-based graphical interface that allows users to experiment with GANs for 2-D data sets with 2-D latent spaces. While being a highly intuitive visualisation tool, the two MLP neural networks in GAN Lab still have many parameters (at least 6 per network, in the simplest case) and are not suited to GAN performance characterisation.

To overcome these disadvantages, our approach is based on (i) low-dimensional scenarios with known probability densities for the data and latent variables; (ii) a simple form for the generator mapping; (iii) a simple form for the discriminator mapping. By judicious choice of the PDFs and mappings in question, and focussing on a least squares cost function as in [6] rather than the logarithmic version in [1], it is possible to construct examples from which analytical results can be derived (without considering systems of ODEs). In section II of the work, we use this approach to provide rigorous counterarguments concerning the optimal discriminator in (1), demonstrating that it is not valid in the case of practical interest where the dimension $n_g$ of the generator input variable $z$ (the latent space) is strictly less than the dimension $n_x$ of the discriminator input variable $x$ (the input data).

In section III, we present a particular low-dimensional LSGAN for which we can explicitly compute the cost function for any point in the parameter space. This example is based on a 1-D latent variable with 1-D data. We show that, for a square law generator of the form $G(z) = gz^2 + h$ with Rayleigh-distributed latent variable $z$ and with a discriminator of the form $D(x) = (1 + \text{erf}(x))/2$, where erf() is the error function, we can obtain accurate representations of the LSGAN cost function for a variety of data distributions, in particular when $x$ is exponentially distributed. The error function has already been applied in probabilistic analyses of neural networks: Amari et al. [20] used it to obtain Fisher information matrix of a single analogue neuron; but it does not seem to have been considered for GANs.

Referring to the example just described as the Rayleigh/Square/Exponential/Erf or R/S/E/E case, we derive a number of theoretically rigorous results concerning the LSGAN cost function (section III-B) and its first and second order derivatives (section IV-A and Appendix). The representations are explicit up to the evaluation of a single, well-behaved, bounded 1-D integral, which can be accurately evaluated numerically by the Monte Carlo method, making the results quasi-analytical and allowing solid conclusions to be drawn. The computations enable an explicit gradient descent (GD) strategy to be realised that is quasi-deterministic given the initial parameter values.\footnote{The use of quasi here indicates that the analysis can be carried out to any degree of accuracy given enough samples in the 1-D MC integral, which has effectively finite support.} The resulting GD algorithm can be realised in various forms, including simultaneous optimisation or, as in conventional GANs, alternating optimisation between discriminator and generator iterations. These ideas are developed further in section IV-B.

It eventuates that the simple 1-D R/S/E/E LSGAN has remarkably complicated behaviour. For certain implementations, the LSGAN does not converge even though the optimal generator, where the GAN output and data distributions are exactly matched, corresponds to a trivial parameter setting. For other implementations we obtain generally slow convergence. Using this simple framework, we provide a counterexample to a claim from the literature: that GANs converge to a saddle point of the cost function [37], i.e. a Nash equilibrium. We explicitly construct an LSGAN that has no saddle points,
and only asymptotic stationary points (maxima, minima and plateaux), for which the discriminator and generator parameters nonetheless converge. Stranger still, the cost function converges to a value that corresponds to perfect discrimination: $E_D(D(x)) = 1$; and perfect generation: $E_D(D(G(z))) = 1$, while not yielding the desired solution: $p_D(x) = p_G(x)$ in (1).

In section V-A, we select two analytical 1-D LSGAN scenarios whose behaviours are typical of those observed in our numerical experiments. For each of these we exhibit the 2-D and 4-D parameter trajectories and cost function evolution under minimax gradient-based optimisation. For the 2-D case, we give examples of the cost surface and its first-order derivatives on which the parameter trajectories are superimposed. We derive in section V-B the explicit stochastic gradient algorithm for the 1-D R/S/E/L SGGAN. This is the same method used in the original GAN formulation to approximate the expectations in the cost function. In section V-C, we provide comparisons between the analytical and stochastic versions of the gradient optimisation to verify the implementations and evaluate their performance. Conclusions and suggestions for further work are covered in section VI.

**II. EXISTENCE OF THE OPTIMAL DISCRIMINATOR**

Proposition 1 in [1], relating to the structure of the optimal discriminator when the generator is fixed, echoed in (1), has a simple and appealing form. In the event that the PDFs of the generator output and the data are equal, the discriminator output, which is always in the interval $[0,1]$, equals one half. To date, this result has been accepted as true in the body of literature concerning generative adversarial networks. The remarkably simple result has an almost equally simple proof, which we paraphrase below.

**Proof:** [1] The optimal discriminator $V[D, G]$ is given by

$$
\int p_D(x) \log[D(x)] dx + \int p_G(z) \log[1 - D(G(z))] dz
$$

$$
= \int (p_D(x) \log[D(x)] + p_G(x) \log[1 - D(x)]) dx
$$

(2)

The function $a \log(y) + b \log(1 - y)$ attains its maximum for $y \in [0, 1]$ at $y = a/(a + b)$, resulting in (1).

The first part of the proof uses a change of variables corresponding to the generator mapping $G(\cdot)$, while the second part computes the functional derivative of $V[D, G]$ with respect to $D(x)$. Applying the transformation of random variables result in the Appendix to (2), we let $f(z) = \log[1 - D(G(z))]$ so that $f(G^{-1}(x)) = \log[1 - D(x)]$ when $x = G(z)$, then:

$$
E_z[f(z)] = \int \log[1 - D(G(z))] p_G(z) dz = E_x[\log(1 - D(x))] = \int \log[1 - D(x)] p_D(x) dx
$$

from which (3) follows, at least formally, when $p_D(x) = p_G(x)$ is the PDF of the generator output. We notice, however, that this result holds under the assumption that $x$ and $z$ have equal dimensions and the transformation $x = G(z)$ is invertible. In the case of GANs, and indeed for any data modelling approach where there are fewer latent variables than data variables, we necessarily have $\text{dim}(x) > \text{dim}(z)$, often by an order of magnitude or more.

A central question is, therefore, what happens when the data dimension exceeds the latent space dimension? To answer this question, we consider some simple, low-dimensional examples where $n_x = \text{dim}(z) = 2$. We assume that $z$ has iid (independent and identically distributed) normal components and consider linear mappings of these components to the output space. We consider the joint data PDF in three possible cases (i) $n_x < n_z$; (ii) $n_x = n_z$, and (iii) $n_x > n_z$.

In the following examples, for vector Gaussian random variables $x$ with mean vector $\mu$ and covariance matrix $P$, we write $x \sim N(\mu, P)$ and adopt $N(x; \mu, P)$ for the corresponding probability density function. Rather than use $p(\cdot)$ to denote a generic PDF, we explicitly subscript PDFs to distinguish them. For compactness, we also use Matlab™ notation; thus $x = [x_1; x_2]$ is the same as $x = [x_1, x_2]^T$, with a similar convention for matrices. The construction of the examples reflects the fact that $G(\cdot)$ is a deterministic function whose domain (or input space) is in a space of lower dimension than its codomain (or output space). This can only be accomplished if there is a deterministic relationship between at least two of the components in the output space.

**Example 1:** Suppose the latent space $z$ is 2-D and the data space $x$ is 3-D. For simplicity, assume that the latent variables are independent with $z_1 \sim N(0, 1)$ and $z_2 \sim N(0, 4)$, so that the joint PDF of the input variables is

$$
p_{z_1, z_2}(z_1, z_2) = \frac{1}{4\pi} \exp\left(-\frac{1}{4}(z_1^2 + z_2^2)\right)
$$

(5)

Consider 3 generator mappings given by

(i) $x_1 = G_1(z) = z_1 + z_2$

(ii) $x_1; x_2 = G_2(z) = [z_1 + z_2; z_1 - z_2]$

(iii) $x_1; x_2; x_3 = G_3(z) = [z_1 + z_2; z_1 - z_2; 2z_2]$

We proceed to compute the joint PDF of $x$ in each of the 3 cases. Case (i): $x_1 = G_1(z)$ is linear so $p_{z_1}(x_1)$ is Gaussian with mean $E(x_1) = E(z_1) + E(z_2) = 0$ and variance $\text{Var}(x_1) = \text{Var}(z_1) + \text{Var}(z_2) = 1 + 4 = 5$. Therefore $p_{z_1}(x_1) = N(x_1; 0, 5)$.

Case (ii): Both $x$ and $z$ are 2-D real random variables related by a linear transformation $x = Az$ where $A = [1, 1; 1, -1]$. According to [38], the PDF of $x = (x_1, x_2)$ is given by

$$
p_{x_1, x_2}(x_1, x_2) = \frac{1}{2}p_{z_1,z_2}(\frac{1}{2}x_1 + \frac{1}{2}x_2, \frac{1}{2}x_1 - \frac{1}{2}x_2)
$$

where the first factor of $\frac{1}{2}$ is from the determinant of the Jacobian of the transformation from $z$ to $x$, leading to

$$
p_{x_1, x_2}(x_1, x_2) = \frac{1}{8\pi} \exp\left[-\frac{1}{2}(x_1, x_2)\begin{bmatrix} 5 & 3 \\ 3 & 5 \end{bmatrix}^{-1}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right]
$$
Case (iii): The transformation from input to output space is:

\[
\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \\ 2 & 0 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}
\]

which can be written as \( x = Tz \) where \( T : \mathbb{R}^2 \rightarrow \mathbb{R}^3 \). The transformation is not invertible since only vectors of the form \((x_1, x_2, x_3)\) where \(x_3 = x_1 + x_2\) can be produced. All 3 components of \( x \) are dependent, so the joint output PDF is most easily obtained by successively conditioning on the variables. There are 3! = 6 ways to order the conditioning and we define (omitting some subscripts for notational convenience)

\[
p_{ijk}(x_1, x_2, x_3) = p(x_1|x_2, x_3)p(x_2|x_3)p(x_3)
\]

where \( i, j \) and \( k \) take values according to the permutation of \((1,2,3)\) in question. In all cases, the first term on the right hand side is a Dirac delta function since \( x_3 = x_1 + x_2 \) for all possible values of the input variables. The remaining two terms can be obtained by straightforward manipulations of the type \( p_{x_1|x_3}(x_1|x_3) = p_{23}(x_1 - \frac{2}{3}x_3) = N(x_1; \frac{3}{2}, 4) \), thus,

\[
p_{x_1|x_3}(x_1|x_3) = \frac{1}{4\sqrt{2\pi}} \exp\left(-\frac{1}{2} \frac{(x_3/2 - x_1)^2}{16}\right)
\]

Combining this reasoning with the results from cases (i) and (ii), we obtain 3 distinct order-dependent joint PDFs \( p_{ijk}(\cdot) = p_{ijk}(\cdot) \) as:

\[
p_{312}(x) = \delta(x_3 - x_1 - x_2) N \left\{ \{x_1; x_2\}; \{0; 0\}, \{5, -3; -3, 5\} \right\}
\]

\[
p_{213}(x) = \delta(x_2 - x_3 + x_1) N \left\{ \{x_1; x_3\}; \{0; 0\}, \{5, 2; 2, 4\} \right\}
\]

\[
p_{123}(x) = \delta(x_1 + x_2 - x_3) N \left\{ \{x_2; x_3\}; \{0; 0\}, \{5, 2; 2, 4\} \right\}
\]

It follows from the preceding example that a change of variable from an input (latent) space to a output (data) space with higher dimension, which necessarily creates deterministic dependencies between some of the output variables, renders the joint PDF of the output space non-unique and degenerate i.e., it contains delta functions. In order to obtain a unique joint PDF for the output space, the ordering of the data variables must be specified. For any ordering, the PDF remains degenerate. From the example, it follows that the number of delta functions in each order-dependent joint PDF is equal to \( \text{dim}(x) - \text{dim}(z) \). As seen, the order dependent joint PDFs are generally not all identical. The next example examines the impact of the PDF degeneracy arising whenever \( n_k > n_z \) on the arguments used in obtaining the optimal discriminator (1).

**Example 2:** Consider the discriminator function \( D(\cdot) : \mathbb{R} \rightarrow \{0, 1\} \) given by \( D(x) = 1 - \exp(-x_1^2 - x_2^2 - x_3^2) \) with latent variable PDF \( \delta \) and generator function \( G(z) \) as in Example 1 (iii). The term \( f(z) = \log(1 - D(G(z))) \) simplifies to \(-2(\frac{1}{2}x_1^2 + x_2^2)\). It is straightforward to show that the expectation of \( f(z) \) with respect to the latent variable is:

\[
E_z f(z) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(z) p_z(z) dz = -14
\]

Under the generator output PDF \( p_{123}(x) \), the expectation \( E_x \log(1 - D(x)) \) with respect to \( x \) is:

\[
- \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x_1^2 + x_2^2 + x_3^2) p_{123}(x) dx
\]

\[
= -2 \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x_1^2 + x_2 + x_3) ^ 2 \times N \left\{ z_2; x_3; \{0; 0\}, \{5, 2; 2, 4\} \right\} dx_2 dx_3
\]

\[
= -2 (E(x_2^2) - E(x_2 x_3) + E(x_3^2))
\]

\[
= -2(5 - 2 + 4) = -14
\]

where the delta function in \( p_{123}(x) \) was used in going from line 1 to 2 above.

In a similar manner, under generator output PDF \( p_{312}(x) \) the expectation \( E_x \log(1 - D(x)) \) with respect to \( x \) is:

\[
- \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x_1^2 + x_2^2 + x_3^2) p_{312}(x) dx
\]

\[
= -2 \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x_1^2 + x_1 + x_2 + x_3) ^ 2 \times N \left\{ z_1; x_2; \{0; 0\}, \{5, -3; -3, 5\} \right\} dx_1 dx_2
\]

\[
= -2 (E(x_1^2) + E(x_1 x_2) + E(x_3^2))
\]

\[
= -2(5 - 3 + 5) = -14
\]

This example demonstrates that \( E_x \log(1 - D(G(z))) = E_x \log(1 - D(x)) \) still appears valid in the particular case, \( n_z = 2, n_x = 3 \), which justifies going from (2) to (3) when \( n_x > n_z \). However, unlike the \( n_z = n_x \) case covered in (4), the astute reader will have noticed that the above integration variables \( x = (x_1, x_2, x_3) \) are the generator outputs and not the data variables. Formally, we are justified in using any variable as a dummy variable in the integral (3) in order to combine the two expectations. Moreover, if we wish to retain the same dimension in the \( x \) integral with respect to the generator output (which is the case here), there are necessarily \( n_x - n_z \) delta functions in the integrand, which is a positive number whenever \( n_z < n_x \).

On the other hand, the final part of Proposition 1 of [1], following equation (3), involves the calculus of variations. This can only be applied to continuously differentiable integrands (in fact, \( C^{n+1} \) where the integrand involves derivatives of order \( n \)). In the GAN case, we require the integrand to be \( C^1 \) with respect to both \( x \) and \( D(\cdot) \). Due to the presence of delta functions in the term \( p_{312}(x) \) in (3) when \( n_z < n_x \), the integrand is not \( C^1 \) in the integration variable \( x \). We therefore conclude that the optimal discriminator cannot be expressed as in (1) when \( n_z < n_x \). Further implications of this observation are discussed in the conclusions section VI.

Evidence of the correctness of our assertion can be deduced from the results in [39], which provides GAN simulations using an ODE solver to perform the parameter estimation. Qin et al. present simulations for a 2-D Gaussian mixture with 32-D latent variable (\( n_z = 32, n_x = 2 \)) and for the CIFAR-10 data set with a 128-D latent variable (\( n_z = 128, n_x = 32 \times 32 \times 3 = 3072 \)). In the first case \( n_z \geq n_x \) and the generator and discriminator losses converge to the
Nash equilibrium values of log(2) and log(4) predicted in [1] (Theorem 1). In the second case, \( n_z < n_x \) and convergence to the predicted values does not occur within 1.20E+6 steps, although the generator loss is close to log(2).

III. 1-D LEAST SQUARES GAN

A. CHOICE OF MODEL

We saw via the counterexample in section II that the optimal discriminator is not defined when \( \text{dim}(x) > \text{dim}(z) \). In this section we examine the \( n_z = n_x \) case when the discriminator has a particular functional form. We are interested in addressing the issue of convergence of the minimax optimisation algorithm used in GANs. We want our model to be as simple as possible to evaluate to high accuracy—preferably an explicit analytical expression in terms of known functions, or, failing this, a rapidly convergent power series or well-behaved Monte Carlo integral. In seeking a low-dimensional analytical example, we quickly turned away from the original form of the cost functional used in the minimax optimisation in [1], namely

\[
\min_G \max_D V[D, G] = E_x \log D(x) + E_z \log[1 - D(G(z))] \tag{6}
\]

where the variational optimisations over the functions \( G(\cdot) \) and \( D(\cdot) \) are replaced by parametric optimisations based on an assumed form for the generator and discriminator mappings (usually based on artificial or deep neural networks). Even in the 1-D case, i.e., when \( n_z = n_x = 1 \), we could not find any nontrivial distributions for which \( V[D, G] \) yielded an explicit form, even when combined with very simple functions for \( D \) and \( G \) such as \( D(x) = 1 - \exp(-(ax + b)) \) and \( G(z) = gz + h \).

Removing the logarithm from the cost function results in a cost functional of the form

\[
V_0[D, G] = E_x D(x) + E_z [1 - D(G(z))] \tag{7}
\]

which resembles the cost function of the Wasserstein GAN, more clearly represented in [40]. Application of the variational argument used in (3) leads to an optimality condition: \( p_d(x) = p_G(x) \), independent of the discriminator function \( D(\cdot) \). A more promising avenue is provided when one considers a variational optimisation based on a sum of squares cost function of the form:

\[
\min_G \max_D V[D, G] = E_x D(x)^2 + E_z [1 - D(G(z))]^2 \tag{8}
\]

which is similar (though not identical) to the cost functionals proposed in [6] for a class of least squares GANs. The least squares cost functional in (7) has the following properties:

1. \( E_x D(x)^2 = 0 \) if the discriminator consistently classifies true data as false (D loses);
2. \( E_x D(x)^2 = 1 \) if the discriminator consistently classifies true data as true (D wins);
3. \( E_z [1 - D(G(z))]^2 = 0 \) if the generator consistently generates data that are indistinguishable from true data (G wins);
4. \( E_z [1 - D(G(z))]^2 = 1 \) if the generator consistently generates data that are classified as false by the discriminator (G loses).

Properties (iii) and (iv) implicitly assume that the discriminator is perfect. It should be clear that, with \( D(x) \in [0, 1] \), the objective function for this least squares GAN (LSGAN) lies in the interval [0, 2]. Noteworthy values of \( V[D, G] \) are: \( V = 0 \) if D loses and G wins; \( V = 1 \) if either (both D and G lose) or (both G and D win); \( V = 0 \) if D loses and G wins; \( V = 2 \) if D wins and G loses. Note that \( V = 1 \) may correspond to conditions other than the two just mentioned.

Applying change of variables and taking the functional derivative with respect to \( D \), on the assumption that \( n_z = n_x \), gives the “optimal discriminator” as:

\[
D^*_G(x) = \frac{p_G(x)}{p_d(x) + p_G(x)} \tag{8}
\]

As in the original GAN (6), \( D^*_G(x) = \frac{1}{2} \) when \( p_G(x) \equiv p_d(x) \). For the remainder of the paper, we consider 1-dimensional \( n_z = n_x = 1 \) LSGANs based on (7).

Turning to the construction of an analytical 1-D LSGAN example, a number of possibilities were explored.

1. 1) Discriminator functions \( D(x) \): logistic function \((1 + \exp(gz))^{-1}\) or error function-based \((1 + \text{erf}(\xi))/2\), where \( \xi = ax + b \).
2. 2) Generator functions \( \hat{g} = G(z) \): linear \(gz+h\) or quadratic \(g^2 + h\).
3. 3) Latent variable PDF \( p_z(z) \): standard Gaussian, exponential, Rayleigh.
4. 4) Input data PDF \( p_d(x) \): Gaussian, exponential, Rayleigh & Maxwell (with different parameters to latent variable PDF).

For brevity, we spare the reader the details of which combinations result in workable analytical models, presenting results only for the R/S/E/E case, defined for \( x \geq 0 \) and \( z \geq 0 \). As explained in section I, the latent variable is Rayleigh, the generator is a square law (plus constant), the data are exponential with PDF \( c \exp(-cx) \) and the discriminator uses erf(). The discriminator parameters are \((a, b)\) and the generator parameters \((g, h)\), with \( c \) a configuration parameter that sets the variance of the data distribution. The choice of an erf-based discriminator is not optimal for exponential data since (8) implies that \( D^*_G(x) \) is a (scaled) logistic function. We note in passing that the case of Gaussian latent variable with linear generator, combined with Gaussian data and erf-based discriminator, or G/L/G/E, is also amenable to analysis using the same methods we subsequently present.

B. 1-D LSGAN COST FUNCTION EVALUATION

Before presenting the main result for the R/S/E/E case of the 1-D least squares GAN, we need a number of integral formulae. These will also be useful for later sections where we calculate the derivatives for gradient descent optimisation. Due to the special form of the LSGAN cost function, we will see that all of the integrals arising in the analysis can be reduced to just one well-behaved integral \( I_4 \) for which numerical evaluation is required. The reason is that the error function, which is the anti-derivative of \( \exp(-x^2) \), forms a “closed family” for integrands on \([0, \infty)\) (with \( m \)
and $n$ non-negative integers) of the form:

$$
\int_0^{\infty} \operatorname{erf}^{n}(ax + b) x^n e^{-c x^2 + 2dx} \, dx, \quad a > 0, \ c \geq 0.
$$

(9)

Integrals in this family are reducible to explicit functions involving exponentials and error functions and, when $m \leq 2$, the single integral $I_5$. Apart from the latter, there are no approximations by way of power series or hypergeometric functions. This is also the reason we did not choose a 1-D GAN discriminator based on the logistic function, for which there are few, if any, closed-form integrals. The sceptical reader may ask why all the required integrals cannot simply be evaluated by Monte Carlo integration. The rationale is simple: for a given precision, minimising the number of required numerical integrations speeds up evaluation of the LSGAN cost function and its derivatives and, at the same time, minimises the numerical errors.

In this and the remaining sections we use $\Phi(x)$ as shorthand for $\operatorname{erf}(x)$. The relevant references for error function integrals consulted in the derivations are [41]–[44]. All the preliminary integrals have a closed form and we maintain the notation from [42] pertaining to the binomial coefficient and Gamma function. Where proofs are not provided, they are either trivial or the integral is covered in one or more of the references. All integrals have been checked either via symbolic algebra software or direct Monte Carlo integration. $Z_+$ denotes the positive integers. All parameters and variables appearing in the formulae are real.

Integral $I_{2r}(q)$ corrects an error in the table contained in [42].

**Definition 1 (Preliminary Integrals):** With $n, r \in Z_+$, $a > 0$, $b = a \beta$, $\beta > 0$,

$$
I_k = \int_0^q u^k e^{-u^2} \, du, \quad q \geq 0, \ k \in Z_+
$$

$$
I_{2r-1}(q) = \frac{1}{2} \Gamma(r) - e^{-q^2} \sum_{j=0}^{r-1} \frac{1}{2} \Gamma(r - j) \frac{q^{2(j-1)}}{2^{r-j}}
$$

$$
I_{2r}(q) = \frac{1}{2} \Gamma(r + \frac{1}{2}) \Phi(q) - e^{-q^2} \sum_{j=0}^{r-1} \frac{1}{2} \Gamma(r + \frac{1}{2}) \frac{q^{2(j-1)}}{2^{r-j}}
$$

$$
T_n(a, b) = \int_0^{\infty} \chi^n e^{-\alpha x^2 + 2bx} \, dx
$$

$$
T_n(a, b) = a^{-n+1} e^{(b/a)^2} \sum_{k=0}^{n} \binom{n}{k} (b/a)^{n-k} \int_{-\beta}^{\infty} u^k e^{-u^2} \, du
$$

$$
\int_{-\beta}^{\infty} u^k e^{-u^2} \, du = \begin{cases} 
I_k(\infty) + \sqrt{\pi} \beta I_k(\beta) & k = 2r \\
I_k(\infty) - \sqrt{\pi} \beta I_k(\beta) & k = 2r - 1
\end{cases}
$$

**Definition 2 (LSGAN-specific Integrals):** With: $n \in Z_+$ and $a, c, s > 0$,

$$
I_9^{(a)}(a, b, c, d) = \int_0^{\infty} \chi^n \Phi(ax + b) e^{-cx^2 + 2dx} \, dx
$$

(10)

$$
\Psi_n^{(a)}(a, b, c, d) = T_n(c, d) - I_9^{(a)}(a, b, c, d)
$$

(11)

$$
\Psi_n^{(a)}(a, b, c, d) = T_n(c, d) + I_9^{(a)}(a, b, c, d)
$$

(12)

$$
I_9(a, b, c, m, s) = \int_0^{\infty} \chi \Phi(ax + b) e^{-\frac{1}{4}(s-m)^2} \, dx
$$

(13)

$$
I_9^{(a)}(a, b, c) = c \int_0^{\infty} \chi^n \Phi(ax + b) e^{-cx} \, dx
$$

(14)

$$
I_9^{(a)}(a, b, c) = c \int_0^{\infty} \chi^n \Phi^2(ax + b) e^{-cx} \, dx
$$

(15)

**Lemma 1 (Reduction formula for $I_9^{(a)}$):**

$$
I_9^{(a)}(a, b, c, d) = \frac{d}{c^2} I_5^{(a)}(a, b, c, d) + \frac{3 - n}{2c^2} I_5^{(a-2)}(a, b, c, d)
$$

$$
+ \frac{ae^{-b^2}}{c^2 \sqrt{\pi}} T_{n-1}(\sqrt{a^2 + c^2}, d - ab), \ n \geq 2
$$

$$
i_9^{(1)}(a, b, c, d) = \frac{1}{2c^2} \Phi(b) + \frac{d}{c^2} I_9^{(a)}(a, b, c, d)
$$

$$
+ \frac{ae^{-b^2}}{c^2 \sqrt{\pi}} T_0(\sqrt{a^2 + c^2}, d - ab)
$$

$$
i_9^{(0)}(a, b, c, d) = \sqrt{\frac{\pi}{c}} \exp\left(\frac{d^2}{c^2}\right) I_9(a, b, d, c, \frac{1}{2c^2})
$$

(16)

Proof: Integrate by parts to obtain $I_9^{(a)}(a, b, c, d)$ as:

$$
\chi^n e^{-cx^2 + 2dx} \Phi(ax + b) e^{-cx^2 + 2dx} \, dx
$$

$$
\chi^n e^{-cx^2 + 2dx} \Phi(ax + b) e^{-cx^2 + 2dx} \, dx
$$

Rearrange to obtain (all $I_5$ integrals have same argument $(a, b, c, d)$):

$$
2c^2 I_5^{(a+2)} = 2d I_5^{(a+1)} + (1 - n) I_5^{(a)}
$$

$$
+ \frac{2a}{\sqrt{\pi}} \int_0^{\infty} \chi^n e^{-cx^2 + 2dx} \, dx
$$

from which the formula for $I_9^{(a)}$ follows.

**Lemma 2 (Evaluation of $I_9^{(a)}$):** Results are only required for $n = 0, 1, 2$, $\beta_1 = -(ab + c/2)$.

$$
i_9^{(2)}(a, b, c) = \frac{2}{c^2} \Phi(b) + \frac{2ae^{-b^2}}{\sqrt{\pi}}
$$

$$
\times \left\{ T_2(a, \beta_1) + \frac{2}{c} T_1(a, \beta_1) + \frac{2}{c^2} T_0(a, \beta_1) \right\}
$$

$$
i_9^{(1)}(a, b, c) = \frac{1}{c} \Phi(b) + \frac{2ae^{-b^2}}{\sqrt{\pi}} \left\{ T_1(a, \beta_1) + \frac{1}{c} T_0(a, \beta_1) \right\}
$$

$$
i_9^{(0)}(a, b, c) = \Phi(b) + \exp\left(\frac{c^2}{2a} + \frac{bc}{a^2}\right) \left(1 - \Phi(b + \frac{c}{2a})\right)
$$
Proof: For $I_9^{(2)}(a, b, c)$ we integrate by parts with $u = \Phi(ax + b)$ to obtain:

$$
I_9^{(2)}(a, b, c) = -\frac{ac}{c^2} \left( \frac{x^2}{c} + \frac{2x}{c^2} + \frac{2}{c^3} \right) \Phi(ax + b) \bigg|_0^\infty
+ \frac{2ac}{\sqrt{\pi}} \int_0^\infty \frac{1}{c} \Phi(ax + b) e^{-\frac{(ax + b)^2}{c}} dx
$$

The result follows, noting that $c > 0$ and in view of the definition of $T_9(a, b)$ in (1). For $I_9^{(1)}(a, b, c)$ we integrate by parts as before:

$$
I_9^{(1)}(a, b, c) = -\frac{ac}{c^2} \left( \frac{x^2}{c} + \frac{2x}{c^2} + \frac{2}{c^3} \right) \Phi(ax + b) \bigg|_0^\infty
+ \frac{2ac}{\sqrt{\pi}} \int_0^\infty \frac{1}{c} \Phi(ax + b) e^{-\frac{(ax + b)^2}{c}} dx
$$

from which the result readily follows. $I_9^{(0)}(a, b, c)$ can also be obtained by integration by parts:

$$
I_9^{(0)} = -\frac{ac}{c^2} \left( \frac{x^2}{c} + \frac{2x}{c^2} + \frac{2}{c^3} \right) \Phi(ax + b) \bigg|_0^\infty
+ \frac{2ac}{\sqrt{\pi}} \int_0^\infty \frac{1}{c} \Phi(ax + b) e^{-\frac{(ax + b)^2}{c}} dx
$$

In the evaluation of the integral on the right hand side, a change of variable to $y = ax + b + c/(2a)$ is required.

Lemma 3 (Evaluation of $I_1^{(n)}$): Results are only required for $n = 0, 1, 2$, $\beta_1 = -(ab + c/2)$.

$$
I_1^{(2)}(a, b, c) = \frac{2}{c^2} \Phi^2(b) + \frac{4a}{\sqrt{\pi}} e^{-b^2} \left( I_5^{(2)}(a, b, a, \beta_1) + \frac{2}{c^2} I_5^{(1)}(a, b, a, \beta_1) + 2 I_5^{(0)}(a, b, a, \beta_1) \right)
$$

$$
I_1^{(1)}(a, b, c) = \frac{1}{c} \Phi^2(b) + \frac{4a}{\sqrt{\pi}} e^{-b^2} \left( I_5^{(1)}(a, b, a, \beta_1) + \frac{1}{c} I_5^{(0)}(a, b, a, \beta_1) \right)
$$

$$
I_1^{(0)}(a, b, c) = \Phi^2(b) + \frac{4a}{\sqrt{\pi}} e^{-b^2} I_5^{(0)}(a, b, a, \beta_1)
$$

Proof: For $I_1^{(2)}(a, b, c)$ we integrate by parts with $u = \Phi^2(ax + b)$ to obtain:

$$
-\frac{ac}{c^2} \left( \frac{x^2}{c} + \frac{2x}{c^2} + \frac{2}{c^3} \right) \Phi^2(ax + b) \bigg|_0^\infty
+ \frac{4ac}{\sqrt{\pi}} \int_0^\infty \frac{1}{c} \Phi(ax + b) e^{-\frac{(ax + b)^2}{c}} dx
$$

Noting that $c > 0$ and the definition of $I_5^{(n)}(a, b, c, d)$ in (10), combining exponentials yields the result in the Lemma.

For $I_1^{(1)}(a, b, c)$ we again integrate by parts and combine exponentials:

$$
I_1^{(1)}(a, b, c) = -\frac{ac}{c^2} \left( \frac{x^2}{c} + \frac{2x}{c^2} + \frac{2}{c^3} \right) \Phi^2(ax + b) \bigg|_0^\infty
+ \frac{4a}{\sqrt{\pi}} \int_0^\infty \frac{1}{c} \Phi(ax + b) e^{-\frac{(ax + b)^2}{c}} dx
$$

The result follows, noting that $c > 0$ and in view of the definition of $T_9(a, b)$ in (1). For $I_1^{(0)}(a, b, c)$ we again integrate by parts and combine exponentials:

$$
I_1^{(0)}(a, b, c) = -\frac{ac}{c^2} \left( \frac{x^2}{c} + \frac{2x}{c^2} + \frac{2}{c^3} \right) \Phi^2(ax + b) \bigg|_0^\infty
+ \frac{4a}{\sqrt{\pi}} \int_0^\infty \frac{1}{c} \Phi(ax + b) e^{-\frac{(ax + b)^2}{c}} dx
$$

With the preceding material in hand, we can now state and prove the main result of this section, which is the evaluation of the 1-D LSGAN cost function. For notational convenience, our definitions of the exponential and Rayleigh distributions differ slightly from the standard ones. (The exponential PDF is usually parametrised by $1/c$, while the standard Rayleigh PDF is $\frac{z}{\sigma^2} e^{-z^2/2\sigma^2}$.) The main result is followed by a corollary concerning the “optimal generator,” for whose parameter settings the generator output PDF exactly matches the PDF of the data.

**Theorem 1 (1-D LSGAN Evaluation):** Consider the 1-D least squares GAN cost function defined for $x \geq 0$ and $z \geq 0$:

$$
V[D, G] = E_x[D(x)^2] + E_z[1 - D(G(z))]
$$

in the R/S/E/E case where the latent variable $z$ is Rayleigh distributed with PDF $p_z(z) = \frac{2z}{\sigma^2} e^{-z^2/2\sigma^2}$, the data are exponential with PDF $p_x(x) = e^{-x}$ with $c > 0$, the generator is $G(z) = \frac{\sigma^2}{2\sigma^2} + h z$ with $g > 0$ and the discriminator is $D(x) = (1 + \text{erf}(ax + b))/2$ with $a > 0$. The parametric version of the cost function is $V[D, G] = J_1(a, b, h; c) + \frac{1}{4} J_2(a, b, h; c)$, where $J_1 \triangleq 4E_z[D(x)^2]$ and $J_2 \triangleq 4E_z[1 - D(G(z))]$ are expressible as

$$
J_1(a, b; c) = 1 + 2I_1^{(0)}(a, b, c) + I_1^{(0)}(a, b, c)
$$

$$
J_2(a, b, h; c) = 1 - 2I_1^{(0)}(a, \eta, 1/g) + I_1^{(0)}(a, \eta, 1/g)
$$

where $\eta = ah + b$ and the integrals $I_1^{(0)}$ and $I_1^{(0)}$ are defined in equations (14) and (15).

Proof: In view of (16) for the R/S/E/E case, $J_1(a, b; c)$ is given by:

$$
1 + 2c \int_0^\infty \Phi(ax + b) e^{-c x} dx + c \int_0^\infty \Phi(ax + b) e^{-c x} dx
$$

which translates directly to (17) via the Definitions 2. For $J_2$ we have

$$
J_2(a, b, h; c) = \int_0^\infty \left( 1 - \Phi(\frac{ag^2 + \eta}{c^2}) \right)^2 \frac{2z}{\sigma^2} e^{-z^2/2\sigma^2} dz
$$
We apply a change of variable $\zeta = g z^2$, under which the Rayleigh PDF becomes an exponential PDF [38]:

$$J_2 = \frac{1}{g} \int_0^\infty (1 - \Phi(a \zeta + \eta))^2 e^{-\zeta / g} d\zeta$$

(20)

which yields

$$J_2 = \frac{1}{g} \int_0^\infty e^{-\zeta / g} d\zeta - 2 \frac{1}{g} \int_0^\infty \Phi(a \zeta + \eta) e^{-\zeta / g} d\zeta + \frac{1}{g} \int_0^\infty \Phi^2(a \zeta + \eta) e^{-\zeta / g} d\zeta$$

from which (18) follows from Definitions 2.

REMARKS

1) Both the generator and discriminator parts of the cost function are reduced to a sum of terms all of which are explicit except for $I_1^{(0)}$, which depends on $I_2^{(0)}$ and thereby on the integral $I_8(a, b, m, s)$ in (13).

2) No closed-form could be found by the authors for $I_8$, so numerical evaluation appears to be required. Note that, if the range of integration is extended to $(-\infty, \infty)$, the integral evaluates to $\Phi((b + ma)/\sqrt{1 + 2a^2 s})$.

3) From the definition, $I_8(a, b, m, s) = \int_0^\infty \Phi(ax + b) N(x; m, s) dx$ where $N()$ is a Gaussian PDF. Since $|\Phi(\chi)| \leq 1$, it follows that $|I_8(a, b, m, s)| \leq 1$.

4) The integrand of $I_8$ decreases rapidly for $x > m + n \sqrt{5}$ where $n$ is the number of standard deviations away from the mean $m$. In practice, the support for the integral can be taken as $[0, m + 5 \sqrt{3}]$, whenever this is non-empty.

5) The last two points guarantee a stable and efficient implementation of $I_8$ using Monte Carlo integration, which works by averaging randomly generated values of the integrand in its support region. A numerical library that includes the error function should be used.

We now state the optimal generator result. To avoid ambiguity, we use $\hat{x}$ rather than $g$ to denote the output of the generator and $p_\xi(\cdot)$ its PDF.

Corollary 1 (Optimal Generator): Consider the 1-D least squares GAN cost function defined in (16) in the R/S/E/E case as defined in Theorem 1. The generator parameter setting that yields the desired solution $p_\xi(x) \equiv p_x(x)$ where $\hat{x}$ is the generator output. For this setting, the LSGAN cost function is given by $I^*(a, b; c) = \frac{1}{2} (1 + I_1^{(0)}(a, b, c))$, where the integral is defined in (15), and the K-L divergence $D_{KL}(p_\xi || p_x) = 0$.

Proof: For the R/S/E/E case, the latent variable PDF is $p_\xi(z) = 2z \exp(-z^2)$ and the data are exponential with PDF $p_x(x) = c \exp(-cx), c > 0$. The PDF of the generator output $\hat{x} = g z^2 + h, g > 0$, can be shown to be (see [38]):

$$p_\xi(y) = \frac{1}{g} \exp\left(-\frac{1}{g}(y - h)\right), \ y \geq h,$$

and zero otherwise

Clearly, the parametrisation $g = 1/c, h = 0$ results in $p_\xi(y) = c \exp(-cy) = p_x(y), \forall y \geq 0$. Hence, trivially, we have

$$D_{KL}(p_\xi || p_x) = -\int_0^\infty \log \left(\frac{p_\xi(y)}{p_x(y)}\right) p_x(y) dy = 0$$

In view of (20), substituting $g = 1/c, h = 0$:

$$J_2(a, b, \frac{1}{c}, 0) = \int_0^\infty (1 - \Phi(a \zeta + b))^2 e^{-\zeta / g} d\zeta$$

(21)

Therefore, replacing $\zeta$ by a dummy variable $x$ and combining integrands:

$$J^* = J_1(a, b, c) + J_2(a, b, \frac{1}{c}, 0) = \frac{1}{4} \int_0^\infty (1 + \Phi(ax + b))^2 + [1 - \Phi(ax + b)]^2 e^{-ax} dx$$

$$= \frac{1}{2} \int_0^\infty e^{-ax} dx + \frac{1}{2} \int_0^\infty \Phi(ax + b)^2 e^{-ax} dx$$

which yields the result in the corollary.

REMARKS

1) The desired solution is reachable in the control theoretic sense, that is, for each $c > 0$, there exists a parameter setting for which it is true.

2) Reachability does not imply optimality: optimising the LSGAN cost function in the minimax sense does not necessarily lead to the optimal generator or to the optimal discriminator.

3) Graphical inspection of $J^*(a, b; c)$ for various values of $c$ seems to indicate that this function does not have a unique maximum in the discriminator parameters.

IV. OPTIMISATION ALGORITHMS

A. FIRST DERIVATIVES OF COST FUNCTION

Given the explicit form of the 1-D LSGAN cost function in Theorem 1, it is straightforward to obtain the first-order partial derivatives with respect to the generator and discriminator parameters $(a, b, g, h)$ by differentiation of the integrand, noting that the limits of integration are constant. This is followed by integration of the resultant expressions, which is facilitated by the "closed family" property (9). The results are summarised in the following Lemma. The same procedure can be used to obtain second-order partial derivatives, which, although considerably more complicated, are still straightforward: the subsequent results have been relegated to the Appendix as they are not required for gradient-based optimisation. The second derivatives are only used in the simulations to ascertain the nature of the convergence points of the gradient algorithms.

Lemma 4 (1-D LSGAN first derivatives): The overall 1-D LSGAN cost function is $J = \frac{1}{4}(J_1 + J_2)$. The component derivatives are provided below, bearing in mind that $\frac{\partial g}{g} = 0$, $\frac{\partial h}{h} = 0$. Refer to definitions (11) and (12). $\zeta = g z^2$.
\[ \eta = ah + b, \beta_1 = -(ab + c/2), \gamma_1 = -(an + 1/(2g)). \]
\[
\frac{\partial J_1}{\partial a} = \frac{4}{\sqrt{\pi}} \int_0^\infty (1 + \Phi(ax + b)) x e^{-(ax+b)^2} c e^{-\xi} dx
\]
\[
= \frac{4ce^{-\beta^2}}{\sqrt{\pi}} \Psi_1^+(a, b, a, \beta_1) \tag{22}
\]
\[
\frac{\partial J_1}{\partial b} = \frac{4}{\sqrt{\pi}} \int_0^\infty (1 + \Phi(ax + b)) e^{-(ax+b)^2} c e^{-\xi} dx
\]
\[
= \frac{4ce^{-\beta^2}}{\sqrt{\pi}} \Psi_0^+(a, b, a, \beta_1)
\]
\[
\frac{\partial J_2}{\partial a} = \frac{-4}{g\sqrt{\pi}} \int_0^\infty (1 - \Phi(a\zeta + \eta)) (\zeta + h) e^{-\zeta/g - (a\zeta + \eta)^2} d\zeta
\]
\[
= \frac{-4e^{-\eta^2}}{g\sqrt{\pi}} \Psi_0^+(a, \eta, a, \gamma_1) + \Psi_0^-(a, \eta, a, \gamma_1) \tag{24}
\]
\[
\frac{\partial J_2}{\partial \eta} = \frac{-4a}{g\sqrt{\pi}} \int_0^\infty (1 - \Phi(a\zeta + \eta)) e^{-\zeta/g} e^{-(a\zeta + \eta)^2} d\zeta
\]
\[
= \frac{-4ae^{-\eta^2}}{g\sqrt{\pi}} \Psi_0^-(a, \eta, a, \gamma_1)
\]
\[
\frac{\partial J_2}{\partial \gamma} = \frac{1}{g^2} \int_0^\infty (1 - \Phi(a\zeta + \eta))^2 (\zeta/g - 1) e^{-\zeta/g} d\zeta
\]
\[
= \frac{1}{g^2} \Omega_1(a, \eta, 1/g - 2) - \frac{1}{g} \Omega_0(a, \eta, 1/g - 2) \tag{26}
\]
where we have defined
\[
\Omega_0(a, b, c, d) = I_{10}^{(n)}(a, b, c) + d I_0^{(n)}(a, b, c)
\]
\[
\frac{\partial J_2}{\partial \gamma} = \frac{1}{g^2} \int_0^\infty (1 - \Phi(a\zeta + \eta))^2 (\zeta/g - 1) e^{-\zeta/g} d\zeta
\]
\[
= \frac{1}{g^2} \Omega_1(a, \eta, 1/g - 2) - \frac{1}{g} \Omega_0(a, \eta, 1/g - 2)
\]
where \(\omega_0(a, b, c, d) = I_{10}^{(n)}(a, b, c) + d I_0^{(n)}(a, b, c)\)
\[
Proof: \quad \frac{\partial}{\partial a} \text{ and } \frac{\partial}{\partial \beta} \text{ follow by completion of the square and the definitions of } T(a, b) \text{ in (1) and } I_0^{(n)} \text{ in (2).}
\]
\[
\frac{\partial}{\partial a} \text{ follows, noting that}
\]
\[
\Psi_1^+(a, \eta, a, \gamma_1) = \int_0^\infty (1 - \Phi(a\zeta + \eta)) e^{-\zeta^2/2 + 2\gamma_1} d\zeta
\]
\[
\Psi_0^-(a, \eta, a, \gamma_1) = \int_0^\infty (1 - \Phi(a\zeta + \eta)) e^{-\zeta^2/2 - 2\gamma_1} d\zeta
\]
with \(\frac{\partial}{\partial g} \text{ and } \frac{\partial}{\partial \gamma} \text{ following in a similar manner. For } \frac{\partial}{\partial g}, \text{ we can show that}
\]
\[
g^2 \frac{\partial^2}{\partial g^2} = \int_0^\infty (\frac{\zeta}{g} - 1) e^{-\zeta/g} d\zeta + 2g I_0^{(0)}(a, \eta, \gamma_1)
\]
\[
- g^2 I_0^{(0)}(a, \eta, \gamma_1) - 2I_1^{(1)}(a, \eta, \gamma_1) + I_0^{(1)}(a, \eta, \gamma_1)
\]
The integral in \(\zeta\) on the right hand side is zero, whence (27) follows in view of definition (28).

\[\Box\]

**B. GRADIENT-BASED OPTIMISATION**

Recalling the variational optimisation problem in (7) for the least-squares GAN, in the 1-D R/S/E/E case, for a fixed exponential data PDF (parameter \(c\)) we wish to minimise the cost function \(J(a, b, g, h; c)\) with respect to the generator parameters \((g, h)\) and maximise \(J\) with respect to the discriminator parameters \((a, b)\). Iterative optimisation is called for since the cost function does not lend itself to a closed-form solution. In the spirit of [1], we apply gradient-based optimisation, making use of the explicit first-order derivatives in section IV-A. Thus, for given initial parameter values, we hope to reach a stationary point of the cost function, provided that one exists. We studied a number of different possible gradient-based optimisation methods. In the following, \(k\) denotes the iteration number, \(\epsilon > 0\) the step-size, \(s\) the sign vector comprising \(+1\) and \(-1\) elements and \(\circ\) the Hadamard (elementwise) product.

- **a** 4-D gradient: 4-D parameter vector \(\theta = [a, b, g, h]^T\) with simultaneous gradient ascent on \((a, b)\) and descent on \((g, h)\) and sign vector \(s = [1, 1, -1, -1]^T\):
\[
\theta(k + 1) = \theta(k) + \epsilon s \circ \nabla_b J(\theta(k)) \tag{29}
\]

- **b** 2-D gradient: fix \(a = a_0\) and \(g = g_0\), then iterate on \((b, h)\):
\[
[b_{k+1}; h_{k+1}] = [b_k; h_k] + \epsilon s \circ \nabla_{(b,h)} J(\theta(k)) \tag{30}
\]

- **c** Alternating 2D+2D: perform alternating cycles of \(N_1\) steps of gradient descent on \((g, h)\):
\[
[g_{k+1}; h_{k+1}] = [g_k; h_k] - \epsilon_1 \nabla_{(g,h)} J(\theta(k)) \tag{31}
\]
where \(\theta(k) = [a_p, b_p, g_k, h_k]^T\) and \((a_p, b_p)\) are the discriminator estimates from the previous (or initial) cycle; followed by \(N_2\) steps of gradient ascent on \((a, b)\):
\[
[a_{k+1}; b_{k+1}] = [a_k; b_k] + \epsilon_2 \nabla_{(a,b)} J(\theta(k)) \tag{32}
\]
where \(\theta(k) = [a_k, b_k, g_p, h_p]^T\) and \((g_p, h_p)\) are the generator estimates from the previous cycle.

- **d** Alternating 1D+1D: fix \(a = a_0\) and \(g = g_0\) and perform alternating cycles of \(N_1\) steps of gradient descent on \(h\):
\[
h_{k+1} = h_k - \epsilon_1 \frac{\partial}{\partial h} J(\theta(k)) \tag{33}
\]
where \(\theta(k) = [a_0, b_0, g_0, h_k]^T\) and \(b_p\) is the discriminator estimate from the previous (or initial) cycle; followed by \(N_2\) steps of gradient ascent on \(b\):
\[
b_{k+1} = b_k + \epsilon_2 \frac{\partial}{\partial b} J(\theta(k)) \tag{34}
\]
where \(\theta(k) = [a_0, b_k, g_0, h_p]^T\) and \(h_p\) is the generator estimate from the previous cycle.

**REMARKS**

1. In the quasi-analytical framework presented here, a single simulation suffices to determine performance of the 1-D LSGAN for a fixed exponential data PDF parameter \(c\) and initial network parameters \((a, b, g, h)\).

2. The only variable parameters are the step size(s) and, for the alternating versions, the number of steps.
This contrasts with conventional studies on GANs where, in addition to various “hyperparameters,” every run of an SGD on the same data set necessarily gives different results due to the use of random sampling to form “minibatches” of the objective function.

3) The original GAN framework in [1] advocates a strategy similar to our alternating 2D+2D optimisation, taking \( N_1 = 1 \) and \( N_2 \geq 1 \).

4) In principle, we could implement the optimisation in the spirit of (7) by performing a complete maximisation or minimisation before alternating. This corresponds to taking both \( N_1 \) and \( N_2 \) very large. This would be similar to the generalised EM algorithm described in [45]. For the simple example we present, however, this strategy does not seem to work since the optimum seems to occur “at infinity.”

5) The framework allows various investigations to be carried out. For instance, the sign vector can be varied to investigate pure ascent or pure descent. This is relevant to the LSGAN in [6], which uses a pure descent strategy.

6) A gradient-based optimisation with \( N \) steps of \( \epsilon \) is not equivalent to carrying out one larger step of size \( N \epsilon \) since the derivative changes at each step.

7) Although the step size can be varied during optimisation (as is often the case in SGD, e.g. using momentum [46] or ADAM [47]), this is not necessary here since the only error in the cost function evaluation is due to Monte Carlo integration, which is insignificant for the examples considered.

8) We could implement the 2-D gradient or the alternating 1D+1D version by fixing \( b \) and/or \( h \) instead of \( a \) and/or \( g \). After examining the form of the cost functions, which do not seem to have stationary points in the latter case, we settled on the recipe above.

9) We implement the alternating 2D+2D optimisation starting with a generator cycle. The iteration can also be implemented starting with a discriminator cycle (as done in [1]). In either case, 4 parameters, \((a, b, g, h)\) are required at initialisation as well as the prior data parameter \( c \).

10) The presence of plateaux in the cost function is not conducive to using second order optimisation techniques (Newton or quasi-Newton), despite having an explicit Hessian matrix.

11) The objective function \( J(a, b, g, h; c) \) can be plotted along with the parameter estimates to ascertain convergence. It is useful to distinguish between convergence of the objective function and convergence (or divergence) of the individual parameter estimates. Examples are given in the following section.

V. ADAPTATION AND CONVERGENCE EXPERIMENTS

Based on the foregoing theoretical background, we are now in a position to construct quasi-analytical numerical examples in the 1-D R/S/E/E case, where the prefix “quasi” implies that accuracy depends on the number of Monte Carlo samples used in the evaluation of integral \( I_8(a, b, m, s) \) in (13). The examples we present are accurate to 3 decimal places using 1.0E+6 samples. While more accuracy can be obtained by using a larger number of samples, we did not notice any differences that affected our conclusions when 1.0E+7 samples were used (apart from the 10-fold increase in simulation time).

As the LSGAN uses the adversarial framework, based on game theory, we would expect the optimisation to attain a saddle point that, according to [1], “is a minimum with respect to [the generator’s] strategy and a maximum with respect to the [discriminator’s] strategy.” To determine whether this plays out as expected, we apply the gradient optimisations in section IV-B. Moreover, we examine whether the 1-D LSGAN converges to the desired solution, namely, when the generator output PDF exactly matches the data PDF as described in Corollary 1.

A. GRADIENT ASCENT/DESCENT

Inspection of cost function surfaces in the 2D case for \( (b, h) \in [-5, 5] \times [-5, 5] \) for many randomly-generated parameter settings \((a, g, c)\), shows there are apparently no saddle points for the R/S/E/E case. There is generally at least one large “flat area” in \((b, h)\) parameter space where \( J(b, h) \approx 1 \) that, as we will see, acts as an attraction region. The presence of the flat area also explains why a Newton algorithm implemented for the 4-D case failed to converge due to non-invertibility of the Hessian. We conjecture that the flatness of this region is related to the saturation of the discriminator function, that is, the large-argument behaviour of the error function.

We selected two of the more interesting examples to present: Case A: \( \{a = 2.2, b = -3, g = 1.5, h = -2, c = 0.5\} \) Case B: \( \{a = 1.21, b = -1, g = 0.35, h = -4, c = 4.04\} \) where \( c \) is the exponential data PDF parameter and \((a, b, g, h)\) specify the initial generator and discriminator parameters. Each example is quasi-deterministic and yields solid conclusions about parameter adaptation in a low-dimensional GAN without the need to perform multiple Monte Carlo runs, avoiding the inevitable variability that this entails. For each case, we present (i) parameter estimates versus iteration plots obtained by gradient optimisation; (ii) cost function (including \( J_1 \) and \( J_2 \)) versus iteration; (iii) cost function surface with parameter trajectory; (iv) cost function derivative surface with respect to \( b \) and \( h \). Plots for (i) and (ii) are presented for the 2-D and 4-D gradient algorithms in section IV-B. For the 2-D case, parameters \( a \) and \( g \) remain fixed. In all cases the step size(s) equal 1 and 1.0E+6 samples are used in the Monte Carlo evaluation of integral \( I_8 \) (13), which is adequate for an accuracy of 3 significant figures. The \( I_8 \) integral is computed twice per iteration only: once for the generator and once for the discriminator.

The alternating versions of these optimisation algorithms were also implemented for \( N_1 = 1 \) and \( N_2 = 1 \) and gave such similar trajectories to their “simultaneous” counterparts.
that the results are not presented. Results for other values of $N_1$ and $N_2$ show an obvious tendency: trading off the generator and discriminator dynamics depending on whether $N_1 > N_2$ or vice versa, and are not presented for brevity.

Results for case A are depicted in Fig. 1. For the 2-D gradient algorithm with $\epsilon = 0.4$ (top left plot), the generator parameter $h$ converges much faster (in about 50 iterations) than the discriminator parameter $b$. At iteration 250, the rate of change of $b$ is $\dot{b} \approx 0.020$. The Hessian matrix at iteration 250 is $[-0.073, 0; 0, 0]$. The dynamics of the discriminator are extremely slow: by iteration 50,000, $\dot{b} = 3.5E - 5$. Cost function evolution is shown in the top right of Fig. 1. The generator cost $J_2$ is starts higher than the discriminator cost $J_1$ but trends downwards and crosses over around iteration 32, stabilising at zero at around iteration 50. The discriminator cost has more complicated behaviour, eventually converging to 1 around iteration 150.

For the 4-D gradient algorithm with $\epsilon = 0.4$ (Fig. 1 lower plots), the generator parameters ($g, h$) converge more rapidly than the discriminator parameters ($a, b$). Parameter $b$ is again the slowest, with $\dot{b} = 0.0246$; at iteration 250. The dynamics of the discriminator are again very slow: $\dot{a} = 1.3E - 6$ and $\dot{b} = 3.6E - 5$ at iteration 50,000. The cost function evolution (lower right) is similar to that of the 2-D case. For both the 2-D and 4-D gradient, we observed that “saturation” of the objective function occurs by iteration 150 while the rate of change of the discriminator parameters is still significant, indicating a lack of sensitivity of $J$ to the latter. For both algorithms, the cost function components converge to $J_1 = 1$ and $J_2 = 0$, corresponding to {both G and D win}. Note that we can resolve the ambiguity in the $J = 1$ case here only because we know the component costs in $J_1 + J_2 = 1$.

A more intuitive explanation of the dynamics of the 1-D R/S/E/E LSGAN is furnished by the cost function surface shown in Fig. 2 along with its $b-$ and $h-$partial derivatives in Figs. 3 and 4 respectively. In case A, the trajectory ($b_k, h_k$) starts its descent down a line of almost constant $b$, with $h$ varying rapidly. A point is reached at a “valley” in the surface where it turns abruptly left and begins to ascend a line of almost constant $h$ until it reaches the flat area, where it eventually comes to rest. It can be seen from the partial derivative surfaces that both derivatives are close to zero at this point.

For varying step sizes $\epsilon \in [0.4, 0.6, 0.8, 1.0]$, with the case A starting point, the end point (at iteration 250) for the generator parameter $h$, which converges rapidly to machine precision, is slightly different: $1.4340, 1.4389, 1.4450, 1.4533$ (computed on 5 runs of each, with absolute errors $\leq 6E - 4$). This means that the parameter trajectory and convergence point for the gradient algorithm depend on the step size—a connection previously noted for training deep convolutional neural networks [48]. Naturally, convergence also depends on the initial point ($b_0, h_0$) for 2-D or 1D+1D, as well as on the number of steps per cycle (for the alternating case), for the same $a, g$ and $c$ values.

Results for case B are depicted in Fig. 5 for a step size of $\epsilon = 0.4$. Once again, only results for the 2D and 4D gradient algorithms are presented, omitting the alternating 2D+2D and 1D+1D versions, which do not differ significantly from
the presented results. For parameter estimation, shown on the left in Fig. 5, the general trend is similar to case A: convergence is faster for the generator parameters than the discriminator parameters in both 2D and 4D gradient algorithms. At iteration 250, \( \dot{b} \approx 0.017 \) and \( \dot{h} \approx -1.2E-4 \) for the 2D algorithm, whereas, for the 4D algorithm, \( \dot{a} \approx 9.0E-4 \), \( \dot{b} \approx 0.016 \), \( \dot{g} \approx -5.0E-6 \) and \( \dot{h} \approx -1.6E-5 \), implying that both 2D and 4D algorithms have equally slow convergence in respect of the \( b \) discriminator parameter. Concerning the cost function plots (right side of Fig. 5), the behaviour is broadly similar, although the cost for the 4D algorithm has a more gradual transition to its steady value. Both algorithms almost attain the maximum cost function value of 2, corresponding to \{D wins, G loses\}. This is followed by a collapse in the cost function value back to 1, where \{D wins & G wins\}.

The cost function surface for the 2-D gradient algorithm in case B is shown in Fig. 6 together with its \( b \)- and \( h \)-partial derivatives in Figs. 7 and 8 respectively. The trajectory \((b_k, h_k)\) follows a line of constant \( h \) up the slope of the cost surface until it slows dramatically near the top before turning sharply to the right. From here it gradually swings down the slope, turning to the left as it nears the plateau region, where it eventually stalls. The partial derivatives are close to zero at this point.

For the 1-D LSGAN with 4-D gradient algorithm we cannot directly visualise the cost function, although we can plot its 2-D projections. The 2D surface plots for any pair of parameters are not static since the surface depends on all 4 parameters \((a, b, g, h)\). Proceeding with the pair \((b, h)\), whose trajectory is plotted on the 2-D cost surface, we note that \((a, g)\) also vary during the optimisation according to Fig. 5 (lower left plot). This creates an interesting visual effect that must be seen to be appreciated (Fig. 9 contains snapshots at 9 different times). Animations of 1-D LSGAN cost surface trajectories have been created and can be accessed online (see [49], [50]).

From these two detailed scenarios, we make some observations in relation to the 1-D R/S/E/E LSGAN model.

1) Contrary to assertions in the literature, the minimax optimisation for an LSGAN does not generally converge to a saddle point of the cost function. We have shown by explicit construction that both the 2D and
4D gradient algorithms for the 1-D R/S/E/E LSGAN, as well as their alternating optimisation counterparts, converge to a flat region in parameter space that has very low curvature, with both gradient and Hessian eigenvalues near zero.

2) Even in the presence of “stochastic shocks,” there is ample evidence to suggest that there is a tendency for the parameter updates to stall in flat regions of the cost function. For all practical purposes, such flat regions are convergence zones for a minimax gradient algorithm. This is borne out by the SGD simulations presented in section V-B.

3) The cost function saturates to a value of 1 well before all of the parameters have converged, indicative of lack of sensitivity to certain parameters (particularly those of the discriminator). Moreover, the $J = 1$ value is indeterminate for the LSGAN: without knowledge of the data distribution, it is not possible to say which of the generator/discriminator pair has the upper hand. $J = 1$ can also correspond to {both G and D win} or {both G and D lose}.

4) In both 2D and 4D cases considered, which have similar dynamics, the generator parameters converge much faster (by a factor of 2 or more) than the discriminator ones. It is possible to accelerate the convergence of the discriminator by applying the alternating optimisation by taking $N_1 = 1$ and $N_2 > 1$. This corroborates the suggestion in [1] to perform multiple discriminator updates in each cycle.

5) In the 2D case, for different initial parameter values $(b, h)$ for fixed $(a, g, c)$, the dynamics of the gradient algorithm depend critically on the starting point on the cost surface. The parameter trajectory and its point of convergence also depend on the step size.

6) In the 4-D case, the desired solution of $g = 1/c = 2.0, h = 0$, corresponding to the optimal generator, is not attained even approximately. In case A, the generator instead converges (with gradients of the order of $1E - 22$ at $50,000$ iterations) to $g = 3.22, h = 0.861$. We were unable to find any settings for which the desired solution was attained. This has important ramifications for the existence of optimal solutions for GANs, since it appears that the 1-D LSGAN here does not have a solution that is optimal in any meaningful sense, such as achieving zero divergence either in the Kullback-Liebler (K-L) or Jensen-Shannon sense.

7) Instead of the minimax criterion, it may be more useful to formulate the optimisation problem as a pure minimum or pure maximum. Based on the geometry of the cost function, these strategies should work for the 1-D LSGAN. Recall that the LSGAN in [6] uses pure descent, the minimum of which corresponds to {G wins, D loses}. This may also be relevant to convergence in the conventional GAN [1], which advocates maximising $E_z \log D(G(z))$ instead of $\min E_z \log(1 - D(G(z)))$.

8) Both case A and B have flat areas corresponding to minima and maxima of the cost function, as well as plateaux. Convergence can potentially occur to any of these flat areas, depending on the initial parameter settings and step size. In case B, if we apply an “early stopping criterion”, e.g. stopping at the first time $J$ is close to stationary (right side plots in Fig. 5) instead of iteration 250, the parameter setting closely achieves $J = 2$, which is optimal for the discriminator.

### B. STOCHASTIC GRADIENT ASCENT/DESCENT

In practice, the data distribution in unknown and the gradient optimisation algorithms in section V-A are not implementable. We are therefore obliged to approximate the expectations in the LSGAN cost function by sample-based Monte Carlo estimate (see [45], p.524). In view of (7), we define

$$J = \frac{1}{N} \sum_{i=1}^{N} D(x_i)^2 + [1 - D(G(z_i))]^2$$  \hspace{1cm} (35)
where $x_i$ and $z_i$, $i = 1, \ldots, N$, are samples from the data and latent variable PDFs, respectively. In the case of the 1-D R/S/E/L GAN, the variables are scalar and samples are generated according to $z_i \sim p_z(z) = 2z \exp(-z^2)$, $x_i \sim p_x(x) = c \exp(-c x)$ with $c > 0$. For a given data PDF parameter $c$, $J = J(a, b, g, h)$, where $(a, b)$ are the discriminator parameters and $(g, h)$ the generator parameters. For stochastic gradient descent, we require the first order derivatives of the sampled cost function. These are straightforward to obtain and are stated without proof below:

$$
\frac{\partial J}{\partial a} = \frac{1}{N} \sum_{i=1}^{N} x_i \Phi^+(ax_i + b) e^{-(ax_i + b)^2} - \frac{g z_i^2 + h}{\sqrt{\pi}} \Phi^-(ag z_i^2 + \eta) e^{-(ag z_i^2 + \eta)^2}
$$

$$
\frac{\partial J}{\partial b} = \frac{1}{N \sqrt{\pi}} \sum_{i=1}^{N} \Phi^+(ax_i + b) e^{-(ax_i + b)^2} - \Phi^-(ag z_i^2 + \eta) e^{-(ag z_i^2 + \eta)^2}
$$

$$
\frac{\partial J}{\partial g} = \frac{-1}{N \sqrt{\pi}} \sum_{i=1}^{N} a g z_i^2 \Phi^-(ag z_i^2 + \eta) e^{-(ag z_i^2 + \eta)^2}
$$

$$
\frac{\partial J}{\partial h} = \frac{-1}{N \sqrt{\pi}} \sum_{i=1}^{N} a \Phi^-(ag z_i^2 + \eta) e^{-(ag z_i^2 + \eta)^2}
$$

where $\eta = ah + b$, and we have defined $\Phi^+(x) = 1 + \Phi(x)$ and $\Phi^-(x) = 1 - \Phi(x)$. The SGD proceeds in the same way as gradient optimisation, replacing $\nabla J(\theta(k))$ by $\nabla J(\theta(k))$, where $\theta$ is the parameter vector (refer to section V-A).

### C. COMPARISON OF ANALYTICAL AND SGD OPTIMISATION

The stochastic gradient minimax algorithm corresponding to the 2D gradient optimisation uses the sample-based estimate of the gradient developed in the preceding section in equation (30). For convenience we refer to this algorithm here as SGD despite it being a minimax optimiser. This SGD algorithm was implemented in the Python programming language and is open source code, available via [51]. We provide in this section the results of comparative performance testing of the SGD and the analytical 2D gradient (2DG) algorithm on the case A scenario. We first describe the multi-run behaviour of the 2DG algorithm, which until this point we have claimed is quasi-deterministic, in the sense defined in section I.

To justify our claims of repeatability, we reran the 2-D gradient algorithm simulation from the same initial point given by the case A parameters in section V-A. We performed 50 Monte Carlo runs of the 2DG algorithm, recording at each run the parameter estimates $(b_k, h_k)$ for $k = 1, \ldots, 250$. For clarification, “Monte Carlo” here refers to the random numbers used in the evaluation of the 1-D integral $I_k$ (13). In all runs, we used 5.0E+6 random samples (to yield high accuracy results) and a step size of 0.4.

![Figure 10](image-url) **FIGURE 10.** Comparison of analytical and SGD case A trajectories for 2-D gradient algorithm.

The results for the 2DG are best appreciated by visualisation, which we have shown in the lower left of Fig. 10. The plots show the overlaying of the 50 2-D parameter trajectories in $(b, h)$ space. There is very little variation in the trajectories. The end point in each run is distinguished by a larger dot. For comparison, the SGD algorithm results for 50 Monte Carlo runs are displayed in the upper left of Fig. 10. Each run used 1000 sample points for the data and latent variables via calls to the python (NumPy) random number generator functions exponential and rayleigh, which were scaled to correspond to the definitions in Theorem 1. For the first-order derivatives, a finite difference approximation was used with a step size of 1E−6. Simultaneous alternating 2-D gradient ascent descent was implemented as per (30). For further details, the reader is referred to the online software repository in [51]. As expected, there is considerable variation in the trajectories, although they cluster well around the trajectories of the 2DG algorithm. The top right plot shows a zoom of the end points at $k = 250$ iterations for both the SGD (with 100 runs) and
analytical 2DG algorithms. The spread in the 2DG end points (in red) is insignificant compared with the SGD (blue points). The lower right plot shows the 2-norm error between the mean SGD and mean 2DG parameter estimates as a function of iteration number. Error spikes around iterations 50 and 100 correspond to the “turn to the left” in the earlier case A plots (Figs. 1 and 2).

The Monte Carlo comparison was also carried out for the case B scenario. Plots for this case appear in Fig. 11. Similar comments as in case A apply here. The errors (lower right plot) for the SGD with respect to the 2DG were highest just after iteration 100, corresponding to the sharp turn to the right in Fig. 6. The “quiescent errors” in both cases A and B from iteration 150 onwards, where the trajectory is converging, were both under 0.01 in Euclidean distance. The results provide verification of the correct functioning of the stochastic gradient minimax procedure with respect to its analytical counterpart, as well as the ability of the latter to accurately predict 1-D R/S/E/E LSGAN performance under stochastic gradient optimisation using error function integrals.

VI. CONCLUSION AND FURTHER WORK
Noting the well recognised numerical difficulties associated with training of generative adversarial networks, and the large number of ad hoc modifications since their introduction in 2014, we set out to examine the theoretical consistency of the GAN framework and investigate its convergence properties. Unlike previous works based either on empirical evidence or continuous-time ODEs, we based our arguments on low-dimensional cases from which direct conclusions can be drawn without recourse to “in the limit” or asymptotic arguments.

In section II, we considered the conditions under which the “optimal discriminator” (Proposition 1 in [1]) can be considered valid. We demonstrated by explicit example that the result is plausible when the dimensions \( n_z \) and \( n_x \) of the latent variable \( z \) and data \( x \) are equal. However, when \( n_z < n_x \), which is usually the case for autoencoders, the PDF of the generator output is non-unique and degenerate, containing \( n_x - n_z \) delta functions. While this does not appear to preclude the validity of equation (3) used in the proof of Proposition 1, the subsequent arguments based on calculus of variations cannot be maintained for non-continuous integrands. Thus, whenever \( \text{dim}(x) > \text{dim}(z) \), the optimal discriminator does not exist. Changing the form of the cost function in the variational optimisation, for instance to a least squares GAN, does not fix this. Naturally, the case where the latent space is of lower dimension than the data space is what makes GANs attractive for practical applications.

The implications of the failure of Proposition 1 for any GAN whose generator network maps a lower dimensional input to a higher dimensional output are significant. Previous justifications and modifications to the GAN framework, such as the Wasserstein GAN [24] and others cited in the introduction, rely on this assumption for some of their technical arguments. Further results in [1], like the global minimum result (equation (6) in [1]), are also restricted in their domain of applicability since they derive directly from Proposition 1. Therefore, despite their “mathiness” [52], preceding technical arguments attempting to justify convergence and optimality of GANs require more stringent preconditions, and, at worst, are erroneous.

In section III, we presented a type of one-dimensional GAN based on a least squares criterion: the 1-D LSGAN, which is similar to the GAN in [6], but uses a minimax optimisation criterion. Seeking explicit analytical solutions for the cost function and its derivatives, we were led to a particular case, which we called the 1-D R/S/E/E LSGAN. This case has a Rayleigh distributed latent variable with a square law generator and exponentially distributed data. The discriminator is based on the error function, mapped to the interval \([0,1]\). We took advantage of the “closed family” property of error function integrals when combined with polynomial and exponential functions, to obtain explicit expressions for the 1-D LSGAN cost function and its derivatives up to and including second order. The resulting expressions are not explicit but are reducible to the computation of a single integral (\( I_k \)) on the positive real line that is a product of an error function and a Gaussian PDF. The integral is bounded on \([-1,1]\) and numerically well behaved. We were subsequently able to perform “quasi-analytical” computations to 3 figure accuracy via Monte Carlo integration using \( 1 \times 10^6 \) random samples.

For the low-dimensional R/S/E/E GAN we constructed, we were able to characterise the optimal generator that achieves the desired solution: where the generator output PDF exactly matches the data PDF. The optimal generator is reachable in the parameter space and is unique. We also noted that the erf-based discriminator is not optimal for exponentially distributed data, but an investigation of this aspect is outside the scope of the present work.

Gradient optimisation algorithms were presented for the 1-D R/S/E/E LSGAN for the case of 1D and 2D parameterisations of the generator and discriminator. These algorithms can be implemented as simultaneous 2D or 4D optimisations or alternating 1D+1D or 2D+2D optimisations. Two numerical examples (cases A and B) were selected to illustrate the properties of the R/S/E/E LSGAN. Both of these were seen to exhibit large flat areas (plateaux) in parameter space that act as attractors for minimax optimisation.

Based on our experiments, we drew the following conclusions for the 4 types of gradient algorithms under test: (i) the 1-D R/S/E/E LSGAN does not appear to have any saddle points; (ii) convergence of all gradient algorithms is frequently to a plateau that is neither a saddle point nor a maximum or minimum of the cost function, but which corresponds to an undesirable solution; (iii) convergence is very slow once the plateau is reached; (iv) the discriminator parameters converge more slowly than those of the generator; (v) for a given data parameter, the convergence point depends not only on the initial parameter values but also on
the step size of the gradient algorithm and the number of steps per cycle in the alternating optimisations; (vi) convergence of the cost function occurs faster than convergence of the parameters, indicating lack of sensitivity to the discriminator parameters; (vii) for the 4D optimisation, the desired solution (i.e. the optimal generator) was not attained for the parameter settings we tested.

Our results provide concrete evidence for various numerical problems noticed in empirical studies, including “saturation” and “vanishing gradient” problems. On the other hand, the saturation is not related to the KL or JS divergence, but is simply a lack of sensitivity in the discriminator part of the cost function. The vanishing gradient is not an indicator of unreliable convergence to a “good optimum” but of reliable convergence to a “bad optimum,” far from the desired solution. We conjecture that the LSGAN in [6], which in fact uses a pure descent strategy, is justified because its optimum corresponds to {G wins, D loses}, which is the main application of GANs as a synthetic data generator. We also uncovered evidence that justifies the application of an early stopping criterion in GAN training to obtain either a better discriminator or a better generator.

Comparative simulations using stochastic gradient minimax optimisation were carried out. These simulations showed good alignment between the analytical and SGD implementations for the 1-D R/S/E/E LSGAN and gave an indication of the relative accuracies of the two methods (for the number of samples used). Moreover, the analytical approach based on error function integrals is a very accurate predictor for the mean SGD parameter trajectories. Since some of the test evidence on which our observations are based is not included, we have made available the software used for the SGD algorithm so that researchers can perform their own experiments (see [51]).

Concerning further work, we mention that our low-dimensional studies are just an initial foray into a vast and mostly unexplored area of understanding, and hence improving, unsupervised approaches to machine learning based on generative adversarial networks. For instance, we have been unable to provide a satisfactory explanation for the classical problems noticed in empirical studies, including “saturation” and “vanishing gradient” problems. On the other hand, the classification framework is the ability to accurately compute the cost function, gradients and Hessian matrix to high accuracy. A useful adjunct would be a rapid and accurate implementation of the gradient algorithm in this type of work. Although the R/S/E/E model we studied appears to have no saddle points, a slightly more complicated model may possess them. Locating saddle points in high dimensions is an interesting problem in itself (see [53] and references therein), and may point to optimisation strategies that naturally converge to saddle points for GANs. Such research would be complementary to existing work on escaping saddle points, which are known to trap gradient descent algorithms [54]. The framework we have presented also provides a path to analyse more complicated and higher dimensional GANs.

APPENDIX

TRANSFORMATION OF RANDOM VARIABLES

Re-expressing (2) as (3) involves a “reparametrization trick” (cf. [55]), i.e., a mapping of the latent variables by some deterministic function \( G(\cdot) : \mathbb{R}^n \to \mathbb{R}^n \). Consider the expectation of a function of a scalar random variable \( z \) with PDF \( p_z(z) \):

\[
E_z[f(z)] = \int_{-\infty}^{\infty} f(z) p_z(z) \, dz
\]

and apply an invertible change of variable by letting \( x = g(z) \) where \( x \) is a scalar random variable with PDF \( p_x(x) \), with the two PDFs satisfying

\[
p_x(x) \, dx = p_z(z) \, dz
\]

It follows that the expectation can be rewritten as

\[
E_z[f(z)] = \int_{-\infty}^{\infty} f(g^{-1}(x)) p_x(x) \, dx = E_x[f(g^{-1}(x))]
\]

The result generalises to (i) many-to-one scalar transformations \( G(\cdot) \) with \( \text{dim}(x) = \text{dim}(z) \); and (ii) vector transformations where \( \text{dim}(x) < \text{dim}(z) \) (see [38]). In case (i), the PDF is composed of a sum of terms each of which corresponds to a root of the equation \( g(z) = 0 \). In case (ii), the two PDFs in (37) are linked via the determinant of the Jacobian. In the case where \( \text{dim}(x) > \text{dim}(z) \), the PDF of \( x \) is non-unique and we are led to the situation in Example 1 case (iii).

SECOND DERIVATIVES OF 1-D LSGAN COST FUNCTION

For brevity, stated without proof. \( H_{ij} = \frac{\partial^2 J}{\partial \theta_i \partial \theta_j}, \theta = (a, b, g, h)^T, \eta = ah + b, \beta_0 = -(c/2 + 2ab), \beta_1 = -(ab + c/2), \gamma_0 = -(2\eta + 1/(2g)), \gamma_1 = -(\eta + 1/(2g)). \)

\[
\frac{\sqrt{\pi}}{2} H_{11} = \frac{c}{\sqrt{\pi}} e^{-2\eta^2} T_2(\sqrt{2a}, \beta_0) - ce^{-b^2} \left( a\Psi_3^+(a, b, a, \beta_1) + b\Psi_2^+(a, b, a, \beta_1) \right) + \frac{e^{-2\eta^2}}{g\sqrt{\pi}} \left\{ T_2(\sqrt{2a}, \gamma_0) + 2hT_1(\sqrt{2a}, \gamma_0) + h^2 T_0(\sqrt{2a}, \gamma_0) \right\} + \frac{1}{g} e^{-\eta^2} \left( a\Psi_3^-(a, \eta, a, \gamma_1) + (2ah + \eta) \Psi_2^-(a, \eta, a, \gamma_1) + (ah + 2\eta) \Psi_1^-(a, \eta, a, \gamma_1) + \eta h^2 \Psi_0^-(a, \eta, a, \gamma_1) \right) \]

\[
\frac{\sqrt{\pi}}{2} H_{22} = \frac{c}{\sqrt{\pi}} e^{-2\eta^2} T_0(\sqrt{2a}, \beta_0)
\]
\[ -e^{-b^2} \left\{ a\Psi_1^2(a, b, a, \beta_1) + b\Psi_0(a, b, a, \beta_1) \right\} + \frac{1}{g} \frac{1}{\sqrt{\pi}} e^{-2\eta^2} T_0(\sqrt{2a}, \gamma_0) + \frac{1}{g} \sqrt{\pi} H_{24}^3 = \frac{a}{g} \frac{1}{\sqrt{\pi}} e^{-2\eta^2} \left\{ a\Psi_1^2(a, \eta, a, \gamma_1) + \eta\Psi_0(a, \eta, a, \gamma_1) \right\} + \frac{a}{g} e^{-\eta^2} \left\{ a\Psi_1^2(a, \eta, a, \gamma_1) + \eta\Psi_0(a, \eta, a, \gamma_1) \right\} H_{13} = \frac{1}{g} \frac{1}{\sqrt{\pi}} e^{-\eta^2} \left\{ -a\frac{1}{g} \frac{1}{2} \Psi_2^2(a, \eta, a, \gamma_1) + \left( \frac{h}{g} - 1 \right) \Psi_1^2(a, \eta, a, \gamma_1) - h\Psi_0(a, \eta, a, \gamma_1) \right\} + \left( \frac{h}{g} - 1 \right) \Psi_1^2(a, \eta, a, \gamma_1) - h\Psi_0(a, \eta, a, \gamma_1) \right\} H_{23} = \frac{1}{g} \frac{1}{\sqrt{\pi}} e^{-\eta^2} \left\{ -a\frac{1}{g} \frac{1}{2} \Psi_2^2(a, \eta, a, \gamma_1) + \left(4a^2h + 2ab\right)\Psi_1^2(a, \eta, a, \gamma_1) + \left(2a^2h + 2abh - 4\right)\Psi_0(a, \eta, a, \gamma_1) \right\} H_{33} \text{is given by} \]

\[ \int_0^\infty \left( e^{-\xi/g} - 4e^{-\xi/g} + 2 \right) e^{-\xi/g} d\xi = 0 \]

N.B. The integral has 9 terms, of which sum to zero: \[ H_{44} = \frac{\partial^2 J}{\partial h^2} = aH_{24} \]

REFERENCES

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in Proc. Adv. Neural Inf. Process. Syst., 2014, pp. 2672–2680.

[2] J. Li, L. Jian, and D. Xu, “Unsupervised representation learning of image-based plant disease with deep convolutional generative adversarial networks,” in Proc. 37th Chin. Control Conf. (CCC), Jul. 2018, pp. 1–16.

[3] K. Zhu, X. Liu, and H. Yang, “A survey of generative adversarial networks,” in Proc. Chin. Autom. Congr., 2018, pp. 2768–2773.

[4] K. Wang, C. Gou, Y. Duan, Y. Lin, X. Zheng, and F.-Y. Wang, “Generative adversarial networks: Introduction and overview,” IEEE/CACSA J. Autom. Sinica, vol. 4, no. 4, pp. 588–598, Sep. 2017.

[5] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, “Generative adversarial networks: An overview,” IEEE Signal Process., vol. 35, no. 1, pp. 53–65, Jan. 2017.

[6] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, “On the effectiveness of least squares generative adversarial networks,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 41, no. 12, pp. 2947–2960, Dec. 2019.

[7] Z. Pan, W. Yu, X. Yi, A. Khan, F. Yuan, and Y. Zheng, “Recent progress on generative adversarial networks (GANs): A survey,” IEEE Access, vol. 7, pp. 36322–36333, 2019.

[8] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, “Improved techniques for training GANs,” 2016, arXiv:1606.03498.

[9] Y.-J. Cao, L.-L. Jia, Y.-X. Chen, N. Lin, C. Yang, B. Zhang, Z. Liu, X.-X. Li, and H.-H. Dai, “Recent advances of generative adversarial networks in computer vision,” IEEE Access, vol. 7, pp. 14985–15006, 2019.

[10] J. Gui, Z. Sun, Y. Wen, D. Tao, and J. Ye, “A review on generative adversarial networks: Algorithms, theory, and applications,” IEEE Trans. Knowl. Data Eng., early access, Nov. 23, 2021, doi: 10.1109/TKDE.2021.3130191.

[11] S. Lala, M. Shady, A. Belyaeva, and M. Liu, “Evaluation of mode collapse in generative adversarial networks,” in Proc. IEEE Conf. High Perf. Extreme Comput., Oct. 2018, pp. 1–9.

[12] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein GAN,” 2017, arXiv:1701.07875.

[13] A. Dash, J. Ye, and G. Wang, “A review of generative adversarial networks (GANs) and its applications in a wide variety of disciplines—from medical to remote sensing,” 2021, arXiv:2110.01442.

[14] D. Saxena and J. Cao, “Generative adversarial networks (GANs): Challenges, solutions, and future directions,” 2020, arXiv:2005.00605.

[15] C. Wang, C. Xu, X. Yao, and D. Tao, “Evolutionary generative adversarial networks,” IEEE Trans. Evol. Comput., vol. 23, no. 6, pp. 921–934, Dec. 2019.

[16] S. Chen, W. Wang, B. Xia, Y. You, P. Peng, Z. Cao, and W. Ding, “CDE-GAN: Cooperative dual evolution-based generative adversarial network,” IEEE Trans. Evol. Comput., vol. 25, no. 5, pp. 986–1000, Dec. 2021.

[17] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in Proc. 13th Int. Conf. Artif. Intell. Statist., 2010, pp. 249–256.

[18] A. M. Saxe, J. L. McClelland, and S. Ganguli, “Exact solutions to the nonlinear dynamics of learning in deep linear neural networks,” 2014, arXiv:1312.6120.

[19] H. Shen, “Towards a mathematical understanding of the difficulty in learning with feedforward neural networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 811–820.

[20] S.-I. Amari, “Any target function exists in a neighborhood of any sufficiently wide random network: A geometrical perspective,” Neural Comput., vol. 32, no. 8, pp. 1–17, 2020.

[21] S.-I. Amari, T. Ozeki, R. Karakida, Y. Yoshida, and M. Okada, “Dynamics of learning in MLP: Natural gradient and singularity revisited,” Neural Comput., vol. 30, no. 1, pp. 1–33, 2018.

[22] S.-I. Amari, Information Geometry and its Applications. Japan: Springer, 2016.

[23] D. Tutsui, “Center manifold analysis of plateau phenomena caused by degeneration of three-layer perceptron,” Neural Comput., vol. 32, no. 4, pp. 1–28, 2018.

[24] M. Arjovsky and L. Bottou, “Towards principled methods for training generative adversarial networks,” in Proc. 5th Int. Conf. Learn. Represent., 2017, pp. 1–17.

[25] R. She, P. Fan, X.-Y. Liu, and X. Wang, “Interpretable generative adversarial networks with exponential function,” IEEE Trans. Signal Process., vol. 69, pp. 3854–3867, 2021.

[26] V. Nagarajan and J. Z. Kolter, “Gradient descent GAN optimization is locally stable,” in Proc. 31st Int. Conf. Neural Inform. Process. Syst., 2017, pp. 5591–5600.

[27] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, “GANs trained by a two time-scale update rule converge to a local Nash equilibrium,” in Proc. Ann. Conf. Neural Info. Proc. Syst., 2017, pp. 1–12.

[28] L. Mchedched, A. Geiger, and S. Nowozin, “Which training methods for GANs do actually converge,” in Proc. Int. Conf. Mach. Learn., 2018, pp. 3481–3490.

[29] M. Braun, Differential Equations and Their Applications. New York, NY: Springer, 1993.

[30] B. Johnson and T. Zhang, “A framework of composite functional gradient methods for generative adversarial models,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 1, pp. 17–32, Jan. 2021.

[31] F. Farnia and A. Ozdaglar, “GANs may have no Nash equilibria,” 2020, arXiv:2002.09124.

[32] S. Sidheenk, A. Aimen, and N. C. Krishnan, “Characterizing GAN convergence through proximal duality gap,” in Proc. 38th Int. Conf. Mach. Learn., 2021, pp. 1–20.
