f-GANs Settle Scores!

Siddarth Asokan
Robert Bosch Center for Cyber-Physical Systems
Indian Institute of Science
Bengaluru - 560012, India
siddartha@iisc.ac.in

Nishanth Shetty
Department of Electrical Engineering
Indian Institute of Science
Bengaluru - 560012, India
nishanths@iisc.ac.in

Aadithya Srikanth
Department of Electrical Engineering
Indian Institute of Science
Bengaluru - 560012, India
srikanth.aadithya@gmail.com

Chandra Sekhar Seelamantula
Department of Electrical Engineering
Indian Institute of Science
Bengaluru - 560012, India
css@iisc.ac.in

Abstract

Generative adversarial networks (GANs) comprise a generator, trained to learn the underlying distribution of the desired data, and a discriminator, trained to distinguish real samples from those output by the generator. A majority of GAN literature focuses on understanding the optimality of the discriminator, typically under divergence minimization losses. In this paper, we propose a unified approach to analyzing the generator optimization through variational Calculus, uncovering links to score-based diffusion models. Considering f-divergence-minimizing GANs, we show that the optimal generator is the one that matches the score of its output distribution with that of the data distribution. The proposed approach serves to unify score-based training and existing f-GAN flavors by leveraging results from normalizing flows, while also providing explanations for empirical phenomena such as the stability of non-saturating GAN losses, or the state-of-the-art performance of discriminator guidance in diffusion models.

1 Introduction

Generative modeling refers to the process of learning the underlying distribution of a given dataset, either with the aim of evaluating the density, or generating new unseen samples from the underlying distribution. Generative adversarial networks (GANs, Goodfellow et al. (2014)) have become one of the most popular frameworks for image generation, owing to lower sampling times and state-of-the-art sample quality (Karras et al., 2020, 2021; Sauer et al., 2022). GANs are a two-player game between a generator network $G\colon \mathbb{R}^d \to \mathbb{R}^n$ and a discriminator network $D\colon \mathbb{R}^n \to \mathbb{R}$. In most GAN settings, $d \leq n$. The generator accepts a noise vector $\boldsymbol{z} \sim p_z$; $\boldsymbol{z} \in \mathbb{R}^d$, typically Gaussian or uniform distributed, and transforms it into a *fake* sample $G(\boldsymbol{z})$, with the push-forward distribution $p_g = G_{\#}(p_z)$. The discriminator accepts an input drawn either from the target distribution, $\boldsymbol{x} \sim p_d$; $\boldsymbol{x} \in \mathbb{R}^n$, or from the output of a generator, and learns a *real versus fake* classifier. The objective is to learn the *optimal generator* — one that can generate realistic samples.

GANs Losses: Divergence-minimizing GANs consider a discriminator to is trained to approximate a chosen divergence measure between p_d and p_g. The generator, on the other hand, minimizes this divergence modeled by the discriminator. For example, the standard GAN (SGAN, Goodfellow et al. (2014)) considers the Jensen-Shannon divergence, while the least-squares GAN (LSGAN, Mao et al. (2017)) models Pearson-χ^2 divergence. Nowozin et al. (2016) generalized the formulation to account for any f-divergence, while Uehara et al. (2016) consider extension to Bregman divergences

Neural Information Processing Systems (NeurIPS) 2023 Workshop on Diffusion Models.

as well. Owing to the training instability of divergence-minimizing GANs on non-overlapping distributions, Arjovsky & Bottou (2017) proposed integral probability metrics (IPM) as a viable alternative. In popular IPM-GANs, such as the Wasserstein GAN (WGAN) Arjovsky et al. (2017), Sobolev GAN Mroueh et al. (2018), or Banach WGAN Adler & Lunz (2018), the discriminator performs the role of a *critic*, and approximates the IPM.

Score Matching: Score matching was originally proposed by Hyvärinen (2005) in the context of independent component analysis. Consider the underlying distribution of the data to be modeled, $p_d(\boldsymbol{x})$. The *(Stein) score* (Liu et al., 2016) is the gradient of logarithm of the density function with respect to the data itself, $\nabla_{\boldsymbol{x}} \ln\left(p_d(\boldsymbol{x})\right)$. It generates a vector field that points in the direction where the data density grows most steeply. In score matching, the score can be approximated by a parametric function $S_\phi^{\mathcal{D}}(\boldsymbol{x})$ obtained by minimizing the Fisher divergence (Cover & Thomas, 2006) $\mathscr{F}(S_\phi^{\mathcal{D}}, p_d) = \frac{1}{2}\,\mathbb{E}_{\boldsymbol{x}\sim p_d}\left[\|S_\phi^{\mathcal{D}}(\boldsymbol{x}) - \nabla_{\boldsymbol{x}} \ln\left(p_d(\boldsymbol{x})\right)\|_2^2\right]$. The output of the trained network is used to generate samples through annealed Langevin dynamics in noise-conditioned score networks (NCSN) (Song & Ermon, 2019). Song et al. (2021b) showed that solving the reverse-time stochastic differential equation (SDE) in diffusion models yields a score-based generative framework, unifying the two domains. Recent approaches aim at either improving the approximation quality of the score network (Song et al., 2020; Ho et al., 2020; Song & Ermon, 2020; Gong & Li, 2021), or improving the discretization of the underlying SDEs (Jolicoeur-Martineau et al., 2021; Karras et al., 2022).

Optimality in GANs: A major research focus in GAN optimization is on the optimality of the discriminator function. While Goodfellow et al. (2014) and Mao et al. (2017) considered a pointwise optimization of the discriminator, Mroueh et al. (2018); Yi et al. (2023) and Asokan & Seelamantula (2023) consider a functional approach, and derived differential equations that govern the optimal discriminator, given the generator. Along another vertical, Pinetz et al. (2018), Stanczuk et al. (2021) and Korotin et al. (2022) showed that, in practical gradient-descent-based training, the optimal discriminator is not attained. However, a similar in-depth analysis of the optimal generator in GANs is lacking. Existing approaches rely on an empirical evaluation of the generator (Zhu et al., 2020), analyze the convergence considering infinite-width network (infinite number of nodes per layer) approximations (Franceschi et al., 2022), or derive constraints on the generator when the generator and discriminator are jointly optimized (Liang, 2021). While in most scenarios, the generator can be linked to minimizing the chosen divergence or IPM, the actual functional optimization has not been thoroughly explored. **What does the closed-form optimization of the generator lead to in f-GANs?** In this paper, this is the gap in literature that we seek to answer.

1.1 Our Contribution

We consider the alternating optimization in various divergence-minimizing and IPM-based GAN formulations, retaining the functional form of the optimal discriminator, and analyze the generator loss function through the lens of *variational calculus*. Considering the family of f-GANs, we show that minimizing the f-divergence results in an optimal generator which, given the optimal discriminator, minimizes the error between the score (the gradient of the log-probability) of the target data distribution, and the score of the generator's push-forward distribution. This permits interpreting the f-GANs as performing score-matching. Owing to the score-matching link, our approach is entitled *ScoreGAN*. As a proof of concept, we validate training ScoreGANs on synthetic Gaussian data. The closest approach to ours is that of Franceschi et al. (2023), who derive similar results in the context of generalizing diffusion models as interacting particle flows.

2 Divergence Minimizing GANs

Nowozin et al. (2016) proposed f-GANs, considering f-divergences of the form: $\mathfrak{D}_f(p_d \| \, p_{t-1}) = \int_{\mathcal{X}} f\left(r_{t-1}(\boldsymbol{x})\right) p_d(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}$, where $f\colon \mathbb{R}_+ \to \mathbb{R}$ is a convex, lower-semicontinuous function over the support $\mathcal{X}$ and satisfies $f(1) = 0$ and $r_{t-1}(\boldsymbol{x})$ is the density ratio $r_{t-1}(\boldsymbol{x}) = \frac{p_d(\boldsymbol{x})}{p_{t-1}(\boldsymbol{x})}$. The optimization is given by $\min_G \left\{\max_D \left\{\mathbb{E}_{\boldsymbol{x}\sim p_d}[T(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{x}\sim p_g}[f^c(T(G(\boldsymbol{z}))]\right\}\right\}$, where $T(\boldsymbol{x}) = g(D(\boldsymbol{x}))$, is the output of the discriminator subjected to activation g, and $D^*(\boldsymbol{x})$ is the optimal discriminator. In practice, the optimization is an alternating one, wherein the discriminator D_t is derived given the generator of the previous iteration, G_{t-1}, and the subsequent generator optimization involves computing G_t, given D_t and G_{t-1}. For simplicity, we denote the push-forward distribution

at iteration t as $p_t(\boldsymbol{x}) = G_{t,\#}(p_z(\boldsymbol{z}))$. Within this formulation, the generator optimization becomes:

$$\mathcal{L}_G^f(G; D_t^*, G_{t-1}) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_d}\left[g(D_t^*(\boldsymbol{x}))\right] - \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_{t-1}}\left[f^c(g(D^*(\boldsymbol{x})))\right], \tag{1}$$

where f^c denotes the Fenchel conjugate of f.

Assume that the generator has not converged, *i.e.,* $p_{t-1}(\boldsymbol{x}) \neq p_d(\boldsymbol{x})$, and that the distribution p_d and p_{t-1} have overlapping support. Then, the following theorem gives us the optimality condition for generator (G_t^*), given D_t^*, such that $p_t(\boldsymbol{x}) = G_{t,\#}^*(p_z) = p_d(\boldsymbol{x})$ for f-GANs.

Theorem 2.1. *Consider the generator loss in f-GANs, given by Equation* (1)*. The **optimal f-GAN generator** satisfies the following score-matching condition:*

$$\underbrace{r_{t-1}(\boldsymbol{x})g'(t)\big|_{t=D_t^*} D_t^{*\prime}(y)\big|_{y=\ln(r_{t-1})}}_{\mathscr{C}(\boldsymbol{x};\, p_d,\, p_{t-1})} \nabla_{\boldsymbol{x}} \left(\ln r_{t-1}(\boldsymbol{x})\right) = \mathbf{0}, \tag{2}$$

where $g'(t)$ denotes the derivative of the activation function with respect to D evaluated at D_t^*, $D_t^{*\prime}(y)$ denotes the derivative of the optimal discriminator function with respect to $y = \ln(r_{t-1}(\boldsymbol{x}))$, evaluated at $\ln(r_{t-1}(\boldsymbol{x}))$ (cf. Table 1, Appendix C.1) and $\boldsymbol{x} = G_t^*(\boldsymbol{z})$, $\boldsymbol{z} \sim p_z$.

The proof is discussed in Appendix C.1. For $\boldsymbol{z}$ such that $\mathscr{C}(\boldsymbol{x};\, p_d, p_{t-1}) \neq 0$, the derived solution can further be simplified to yield the score matching condition: $\nabla_{\boldsymbol{x}} \ln\left(p_{t-1}(\boldsymbol{x})\right)\big|_{\boldsymbol{x}=G_t^*(\boldsymbol{z})} = \nabla_{\boldsymbol{x}} \ln\left(p_d(\boldsymbol{x})\right)\big|_{\boldsymbol{x}=G_t^*(\boldsymbol{z})}$. Although the result shows that all f-GAN generators are inherently score-matching in nature, the effect of $\mathscr{C}$ accounts for the difference in training stability observed across f-GAN variants. We discuss these results in Appendix C.1. For example, $\mathscr{C}$ is unity only for reverse-KL (RKL) GANs, *i.e.,* the generator goes to zero only when the scores match exactly. This is consistent with empirical results by Nguyen et al. (2017); Shannon et al. (2020), where the relatively stabler non-saturating GAN loss considered by Goodfellow et al. (2014) was shown to approximate an RKL loss in practice. Interestingly, the analysis carried out by Franceschi et al. (2023), analyzing diffusion models as interacting particle flows, gives rise to optimality conditions that are consistent with Theorem 2.1. In a way, the derived results *close the loop* between GANs and diffusion models. The general solution to both GAN generator training and interacting particle flows is score matching!

Interpreting the Optimal Generator: The optimality condition in f-GANs brings to light the underlying link between f-GANs and score-based models. While NCSN and its variants rely on Langevin dynamics to model data transformations, the optimal generator in GANs can be interpreted as approximating these iterations one-shot. The results derived provide an analytical equivalence between the sampling iterations in score-based diffusion, and the training iterations of a GAN generator. GAN training transforms the generator distribution in the score-matching sense, akin to the iterations in Langevin sampling. Given the link to score-matching, the f-GAN discriminator gradient can be viewed as serving the role of a proxy for the score (cf. Appendix D). This validates the empirical success of discriminator guidance in state-of-the-art diffusion models (Kim et al., 2023).

Practical Considerations: In practice, as the score is undefined when either $p_d(\boldsymbol{x})$ or $p_{t-1}(\boldsymbol{x})$ are zero, the optimality condition cannot be met pointwise, but must be approximated. We therefore consider a least-squares cost. Given a neural network generator G_{θ_t}, where θ_t denotes the network parameters at time t, this gives rise to the *Fisher divergence* between the scores:

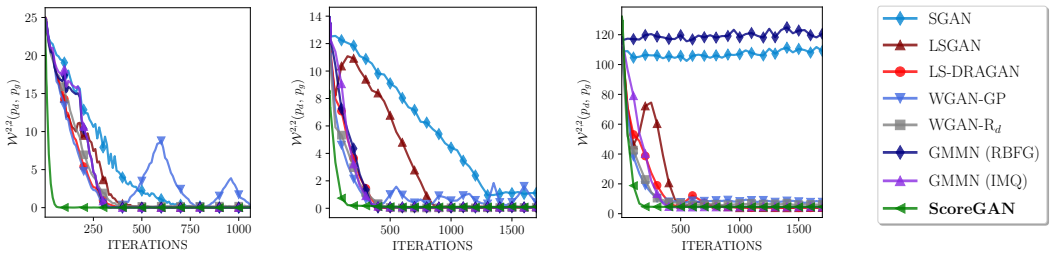
$$\mathcal{L}_G^{\mathrm{Sc}}(\theta) = \mathop{\mathbb{E}}_{\boldsymbol{z}\sim p_z}\left[\|\nabla_{\boldsymbol{x}} \ln(p_{t-1}(\boldsymbol{x})) - \nabla_{\boldsymbol{x}} \ln(p_d(\boldsymbol{x}))\|_2^2\big|_{\boldsymbol{x}=G_{\theta_t}(\boldsymbol{z})}\right],$$

where $\theta^* = \arg\min_\theta \mathcal{L}_G^S(\theta)$. Owing to the score-based approach to training the generator, the proposed approach is called *ScoreGAN*. The above loss involves computing two key terms: (i) The score of the target data; and (ii) The score of the generator distribution. For parametric distributions such as Gaussians, the score of the data can be computed by means of automatic differentiation (Abadi et al., 2016; Paszke et al., 2019). In the case of image data, a pre-trained score network $S_\phi^{\mathcal{D}}$ can be used to approximate the score of the data (Song & Ermon, 2020; Song et al., 2021a; Rombach et al., 2022). To compute the score of the generator, when the dimensionality of the data is relatively small, say $\mathcal{O}(10^3)$, the *change of variables* formula can be used, yielding the following result:

Lemma 2.2. *Consider the push-forward generator distribution $p_t(\boldsymbol{x}) = G_{\theta_t,\#}(p_z)$, where $p_z = \mathcal{N}(\boldsymbol{z}; \mu_z, \Sigma_z)$ and $G_{\theta_t}\colon \mathbb{R}^n \to \mathbb{R}^n$. Then, the generator score is given by:*

$$\nabla_{\boldsymbol{x}} \ln p_t(\boldsymbol{x})\big|_{\boldsymbol{x}=G_{\theta_t}(\boldsymbol{z})} = -\mathrm{J}_{G_{\theta_t}}^{-\mathrm{T}}\left(\nabla_{\boldsymbol{z}} \ln|\det \mathrm{J}_{G_{\theta_t}}(\boldsymbol{z})| + \boldsymbol{z}\right),$$

where $\mathrm{J}_{G_{\theta_t}}$ denotes the Jacobian of the generator G_{θ_t}.



(a) (b) (c)

Figure 1: (Color online) Comparisons between ScoreGAN and the baselines in terms of the Wasserstein-2 distance $\mathcal{W}^{2,2}(p_d, p_g)$ on learning (a) a 2-D; (b) a 16-D; and (c) a 128-D Gaussian. ScoreGAN converge an order of magnitude faster than the baseline GANs on 2-D Gaussians. As the data dimensionality increases, ScoreGAN continue to outpace the baselines.

The proof is discussed in Appendix C.4. Generalizations considering $G\colon \mathbb{R}^d \to \mathbb{R}^n;\ d \ll n$ are discussed in Appendix C.5. In very high dimensions, the Jacobian computations are inefficient, and one could consider training a second score network, S^G_ψ, to approximate the score of the generator, trained jointly with the generator in a *non-adversarial* fashion. As part of this workshop submission, we present experiments on Gaussian data, deferring the analysis of the alternatives to future extensions.

3 Experimental Validation

To validate the observations made in Sections 2, we consider synthetic experiments on learning Gaussians. While these experiments are not targeted towards outperforming state-of-the-art GANs (Sauer et al., 2022; Kang et al., 2023), they serve to illuminate the training dynamics present in GAN variants, and their links to modeling the *score*. As baselines, we consider SGAN (Goodfellow et al., 2014), LSGAN (Mao et al., 2017) and WGAN-GP (Gulrajani et al., 2017), gradient-regularized variants such as LS-DRAGAN (Kodali et al., 2017), WGAN-R$_d$ (Mescheder et al., 2018), and kernel-based generative moment matching networks (GMMNs) with the inverse multi-quadric (IMQ) and Gaussian (RBFG) kernels (Li et al., 2015). We present results on learning 2-, 16-, and 128-dimensional univariate Gaussians. The generator is a linear transformation $\boldsymbol{x} = \boldsymbol{A}\boldsymbol{z} + \boldsymbol{b}$. Additional network details are given in Appendix D. From Figure 1, we observe that in all three scenarios, GMMNs converges faster than the baseline GANs owing to the lack of adversarial training. On low-dimensional 2-D Gaussian learning, ScoreGAN converges the fastest. As the dimensionality increases, the convergence SGAN and GMMN (RBFG) fail to converge on 128-D data, owing to vanishing gradients. To showcase the instability of f-divergence based GANs when p_{t-1} and p_d possess non-overlapping supports, we present results on learning Gaussian mixture data. Similar empirical observations were made when training SGAN on the Dirac measure (Arjovsky & Bottou, 2017). We consider the two-component Gaussian mixture $p_d(\boldsymbol{x}) = \frac{1}{5}\mathcal{N}(-5\mathbf{1}, \mathbb{I}) + \frac{4}{5}\mathcal{N}(5\mathbf{1}, \mathbb{I})$. Figures comparing the generator and data distributions are provided in Appendix D.1. The results indicate that, while the IPM-based GANs converge accurately to the desired target distribution, SGAN misses the less-represented mode located at $\boldsymbol{\mu} = -5\mathbf{1}$. This can be explained through Theorem 2.1 – When the generated samples are far from the data, $p_d(G_{\theta_t}(\boldsymbol{z})) \to 0$, leading to small gradients induced by the rapidly decaying score.

4 Discussions and Conclusion

In this paper, we proposed a novel approach to analyzing the optimal generator in divergence-minimizing through the perspective of variational Calculus. While our analysis covers most popular f-GAN flavors, the analysis can be extended to other closely-related GAN loss function (cf. Appendix C.3). Theorem 2.1 show that in all f-GANs, the generator is a score-matching network. These results deepen our understanding of the optimality in GANs. For examples, the score loss in f-GANs help explain their poor performance on non-overlapping distributions. These insights, together with diffusion-based formulations derived by (Franceschi et al., 2023) provide a framework for deriving equivalent diffusion models, given a GAN, and *vice versa.* Training a second score network, S^G_ψ, to approximate the score of the generator, or analyzing IPM-GAN costs within this framework, are promising directions for future research.

Acknowledgements

Siddarth Asokan is supported by the Microsoft Research Ph.D. Fellowship, the Qualcomm Innovation Fellowship 2023, and the Robert Bosch Center for Cyber-Physical Systems Ph.D. Fellowship. Nishanth Shetty is supported by the Prime Minister's Research Fellowship and the Qualcomm Innovation Fellowship 2023.

References

Abadi, M. et al. TensorFlow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint, arXiv:1603.04467*, Mar. 2016. URL https://arxiv.org/abs/1603.04467.

Adler, J. and Lunz, S. Banach Wasserstein GAN. In *Advances in Neural Information Processing Systems 31*, pp. 6754–6763. 2018.

Ansari, A. F., Ang, M. L., and Soh, H. Refining deep generative models via discriminator gradient flow. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Zbc-ue9p_rE.

Arbel, M., Korba, A., Salim, A., and Gretton, A. Maximum mean discrepancy gradient flow. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Arjovsky, M. and Bottou, L. Towards principled methods for training generative adversarial networks. *arXiv preprints, arXiv:1701.04862*, 2017. URL https://arxiv.org/abs/1701.04862.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 214–223, 2017.

Asokan, S. and Seelamantula, C. S. Euler-Lagrange analysis of generative adversarial networks. *Journal of Machine Learning Research (JMLR)*, pp. 1–100, 2023.

Cover, T. and Thomas, J. *Elements of Information Theory*. Wiley-Interscience, 2006.

Dinh, L., Krueger, D., and Bengio, Y. NICE: non-linear independent components estimation. In *3rd International Conference on Learning Representations, Workshop Track Proceedings*, 2015. URL http://arxiv.org/abs/1410.8516.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real NVP. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=HkpbnH9lx.

Ferguson, J. A brief survey of the history of the calculus of variations and its applications. *arXiv preprint, arXiv:math/0402357*, Feb. 2004. URL https://arxiv.org/abs/math/0402357.

Franceschi, J.-Y., De Bézenac, E., Ayed, I., Chen, M., Lamprier, S., and Gallinari, P. A neural tangent kernel perspective of GANs. In *Proceedings of the 39th International Conference on Machine Learning*, Jul 2022.

Franceschi, J.-Y., Gartrell, M., Santos, L. D., Issenhuth, T., de Bzenac, E., Chen, M., and Rakotomamonjy, A. Unifying gans and score-based diffusion as generative particle models. *arXiv preprint, arXiv:2305.16150*, abs/2305.16150, 2023. URL https://arxiv.org/abs/2305.16150.

Gel'fand, I. M. and Fomin, S. V. *Calculus of Variations*. Prentice-Hall, 1964.

Glaser, P., Arbel, M., and Gretton, A. KALE flow: A relaxed KL gradient flow for probabilities with disjoint support. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Goldstine, H. H. *A History of the Calculus of Variations from the 17th Through the 19th Century*. Springer, New York, 1980.

Gong, W. and Li, Y. Interpreting diffusion score matching using normalizing flow. In *ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models*, 2021. URL https://openreview.net/forum?id=jxsmOXCDv9l.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. C., and Bengio, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, pp. 2672–2680. 2014.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of Wasserstein GANs. In *Advances in Neural Information Processing Systems 30*, pp. 5767–5777. 2017.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *arXiv preprint, arXiv:2006.11239*, 2020. URL https://arxiv.org/abs/2006.11239.

Hyvärinen, A. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(24):695–709, 2005. URL http://jmlr.org/papers/v6/hyvarinen05a.html.

Jolicoeur-Martineau, A., Li, K., Piché-Taillefer, R., Kachman, T., and Mitliagkas, I. Gotta go fast with score-based generative models. In *The NeurIPS DLDE-I Workshop*, 2021. URL https://openreview.net/forum?id=gEoVDSASC2h.

Jordan, R., Kinderlehrer, D., and Otto, F. The variational formulation of the Fokker–Planck equation. *SIAM journal on mathematical analysis*, 29(1):1–17, 1998.

Kang, M., Zhu, J.-Y., Zhang, R., Park, J., Shechtman, E., Paris, S., and Park, T. Scaling up GANs for text-to-image synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Karras, T., Aittala, M., Hellsten, J., Laine, S., Lehtinen, J., and Aila, T. Training generative adversarial networks with limited data. In *Advances in Neural Information Processing Systems 33*, 2020.

Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Karras, T. et al. Alias-free generative adversarial networks. In *Advances in Neural Information Processing Systems*, volume 34, June 2021.

Kim, D., Kim, Y., Kwon, S. J., Kang, W., and Moon, I. Refining generative process with discriminator guidance in score-based diffusion models. In *Intl. Conf. on Machine Learning*, 2023.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, 2015.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. *arXiv preprint, arXiv:1807.03039*, abs/1807.03039, 2018. URL https://arxiv.org/abs/1807.03039.

Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. Improved variational inference with inverse autoregressive flow. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2016.

Kodali, N., Abernethy, J. D., Hays, J., and Kira, Z. On convergence and stability of GANs. *arXiv preprint, arXiv:1705.07215*, May 2017. URL http://arxiv.org/abs/1705.07215.

Korotin, A., Kolesov, A., and Burnaev, E. Kantorovich strikes back! Wasserstein GANs are not optimal transport? In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

Kwon, D., Fan, Y., and Lee, K. Score-based generative modeling secretly minimizes the Wasserstein distance. In *Advances in Neural Information Processing Systems*, 2022.

Li, Y., Swersky, K., and Zemel, R. Generative moment matching networks. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 1718–1727, Jul 2015.

Liang, T. How well generative adversarial networks learn distributions. *Journal of Machine Learning Research*, 22(228):1–41, 2021. URL http://jmlr.org/papers/v22/20-911.html.

Liu, Q. Stein variational gradient descent as gradient flow. In *Advances in Neural Information Processing Systems*, 2017.

Liu, Q., Lee, J., and Jordan, M. A kernelized Stein discrepancy for goodness-of-fit tests. In *Proceedings of The 33rd International Conference on Machine Learning*, Jun 2016.

Mao, X., Li, Q., Xie, H., Lau, R. Y. K., Wang, Z., and Smolley, S. P. Least squares generative adversarial networks. In *Proceedings of International Conference on Computer Vision*, 2017.

Mescheder, L., Geiger, A., and Nowozin, S. Which training methods for GANs do actually converge? In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 3481–3490, Stockholmsmassan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Mroueh, Y. and Nguyen, T. On the convergence of gradient descent in GANs: MMD GAN as a gradient flow. In Banerjee, A. and Fukumizu, K. (eds.), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, Apr 2021.

Mroueh, Y. and Rigotti, M. Unbalanced Sobolev descent. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Mroueh, Y., Li, C., Sercu, T., Raj, A., and Cheng, Y. Sobolev GAN. In *Proceedings of the 6th International Conference on Learning Representations*, 2018.

Mroueh, Y., Sercu, T., and Raj, A. Sobolev descent. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, Apr 2019.

Nguyen, T., Le, T., Vu, H., and Phung, D. Dual discriminator generative adversarial nets. volume 30, 2017.

Nowozin, S., Cseke, B., and Tomioka, R. f-GAN: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems 29*, pp. 271–279. 2016.

Papamakarios, G., Pavlakou, T., and Murray, I. Masked autoregressive flow for density estimation. In *Advances in Neural Information Processing Systems 30*. 2017.

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 2021. URL `http://jmlr.org/papers/v22/19-1028.html`.

Paszke, A. et al. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, volume 32, 2019.

Petersen, K. B., Pedersen, M. S., et al. The Matrix Cookbook. *Technical University of Denmark*, 7 (15):510, 2008.

Pinetz, T., Soukup, D., and Pock, T. What is optimized in Wasserstein GANs? In *Proceedings of the 23rd Computer Vision Winter Workshop*, 02 2018.

Rezende, D. J. and Mohamed, S. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, pp. 15301538, 2015.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Sauer, A., Schwarz, K., and Geiger, A. StyleGAN-XL: scaling StyleGAN to large diverse datasets. volume abs/2201.00273, 2022. URL `https://arxiv.org/abs/2201.00273`.

Shannon, M., Poole, B., Mariooryad, S., Bagby, T., Battenberg, E., Kao, D., Stanton, D., and Skerry-Ryan, R. Non-saturating GAN training as divergence minimization. *arXiv preprint arXiv:2010.08029*, 2020.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. In *Intl. Conf. on Learning Representations*, 2021a. URL https://openreview.net/forum?id=St1giarCHLP.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems*, 2019.

Song, Y. and Ermon, S. Improved techniques for training score-based generative models. In *Advances in Neural Information Processing Systems 33*, 2020.

Song, Y., Garg, S., Shi, J., and Ermon, S. Sliced score matching: A scalable approach to density and score estimation. In *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, volume 115, pp. 574–584, Jul 2020.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=PxTIG12RRHS.

Stanczuk, J., Etmann, C., Kreusser, L. M., and Schnlieb, C.-B. Wasserstein GANs work because they fail (to approximate the Wasserstein distance). *arXiv preprint, arXiv:2103.01678*, abs/2104.11222, 2021. URL https://arxiv.org/abs/2103.01678.

Su, J. and Wu, G. f-VAEs: Improve VAEs with conditional flows. *arXiv preprint, arXiv:1809.05861*, abs/1809.05861, 2018. URL https://arxiv.org/abs/1809.05861.

Uehara, M., Sato, I., Suzuki, M., Nakayama, K., and Matsuo, Y. Generative adversarial nets from a density ratio estimation perspective. *arXiv preprint, arXiv:1610.02920*, abs/1610.02920, 2016. URL https://arxiv.org/abs/1610.02920.

Yi, M., Zhu, Z., and Liu, S. Monoflow: Rethinking divergence GANs via the perspective of differential equations. *arXiv preprint, arXiv:2302.01075*, abs/2302.01075, 2023. URL https://arxiv.org/abs/2302.01075.

Zhu, B., Jiao, J., and Tse, D. Deconstructing generative adversarial networks. *IEEE Transactions on Information Theory*, 66, 2020.

Appendix

Table of Contents

A Mathematical Preliminaries

Consider a vector $\boldsymbol{z} = [z_1, z_2, \ldots, z_n]^{\mathrm{T}} \in \mathbb{R}^n$ and the generator $G : \mathbb{R}^n \to \mathbb{R}^n$, *i.e.,* $G(\boldsymbol{z}) = [G^1(\boldsymbol{z}), G^2(\boldsymbol{z}), \ldots, G^n(\boldsymbol{z})]$. The notation $\nabla_{\boldsymbol{z}} G(\boldsymbol{z})$ represents the gradient matrix associated with the generator, with entries consisting of the partial derivatives of the entries of G with respect to the entries of $\boldsymbol{z}$:

$$\nabla_{\boldsymbol{z}} G(\boldsymbol{z}) = \begin{bmatrix} \frac{\partial G^1}{\partial z_1} & \frac{\partial G^2}{\partial z_1} & \cdots & \frac{\partial G^n}{\partial z_1} \\ \frac{\partial G^1}{\partial z_2} & \frac{\partial G^2}{\partial z_2} & \cdots & \frac{\partial G^n}{\partial z_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial G^1}{\partial z_n} & \frac{\partial G^2}{\partial z_n} & \cdots & \frac{\partial G^n}{\partial z_n} \end{bmatrix}.$$

The Jacobian J can be thought of as *measuring* the transformation that the function imposes locally near the point of evaluation, and is is defined to be the transpose of the gradient, *i.e.,* $\nabla_{\boldsymbol{z}} G(\boldsymbol{z}) = \mathrm{J}_G^{\mathrm{T}}(\boldsymbol{z})$.

Calculus of Variations: Our analysis centers around deriving the optimal generator in the functional sense, leveraging the *Fundamental Lemma of the Calculus of Variations* (Goldstine, 1980; Ferguson, 2004). Consider an integral cost $\mathcal{L}$, to be optimized over a function h:

$$\mathcal{L}\left(h, h'\right) = \int_{\mathcal{X}} \mathcal{F}\left(\boldsymbol{x}, h(\boldsymbol{x}), h'(\boldsymbol{x})\right) \, \mathrm{d}\boldsymbol{x}\,, \tag{3}$$

where h is assumed to be continuously differentiable or at least possess a piecewise-smooth derivative $h'(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathcal{X}$. If $h^*(\boldsymbol{x})$ denotes the optimum, The *first variation* of $\mathcal{L}$, evaluated at h^*, is defined as the derivative $\delta\mathcal{L}(h^*; \eta) = \frac{\partial \mathcal{L}_\epsilon(h^*)}{\partial \epsilon}$ evaluated at $\epsilon = 0$, where $\mathcal{L}_\epsilon(h^*)$ denotes an ϵ-perturbation of the argument h about the optimum h^*, given by

$$\mathcal{L}_{h,\epsilon}(\epsilon) = \mathcal{L}\left(h^*(\boldsymbol{x}) + \epsilon\,\eta(\boldsymbol{x}), h^{*\prime}(\boldsymbol{x}) + \epsilon\,\eta'(\boldsymbol{x})\right)$$

where, in turn, $\eta(\boldsymbol{x})$ is a family of *perturbations* that are compactly supported, infinitely differentiable functions, and vanishing on the boundary of $\mathcal{X}$. Then, the optimizer of the cost $\mathcal{L}$ satisfies the following first-order condition:

$$\left.\frac{\partial \mathcal{L}_{h,\epsilon}(\epsilon)}{\partial \epsilon}\right|_{\epsilon=0} = 0$$

Another core concept in deriving functional optima is the *Fundamental Lemma of Calculus of Variations*, which states that, if a function $g(\boldsymbol{x})$ satisfies the condition

$$\int_{\mathcal{X}} g(\boldsymbol{x})\, \eta(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} = 0$$

for all compactly supported, infinitely differentiable functions $\eta(\boldsymbol{x})$, then g must be identically zero almost everywhere in $\mathcal{X}$. Together, these results are used to derive the condition that the optimal generator transformation satisfies, within various GAN formulations.

B An Overview of Related Works

Links between diffusion and flows can be traced back to the work of Jordan et al. (1998), where the Fokker-Planck equation was shown to lead to a Kullback-Leibler (KL) flow, discretized to give rise to the Langevin Monte Carlo algorithm. However, an analysis under **KL flow** or **Stein flow** (Liu, 2017) for GANs is infeasible, as this requires the analytical form of the target density. Recently, Gong & Li (2021) showed that diffusion score matching can be interpreted as normalizing flows. Our results, in a similar vein, link GAN generator optimization to both flows, and score matching. Mroueh & Nguyen (2021) leverage **MMD flow** (Arbel et al., 2019) to analyze the convergence in MMD-GANs. Recently, Kwon et al. (2022) showed that the score-matching networks in fact solve for the **Wasserstein flow** between p_d and p_g.

The closest approaches to ours are that of **MonoFlow**, proposed by Yi et al. (2023), and the **interacting particles framework** of Franceschi et al. (2023). In MonoFlow, Yi et al. (2023) showed that the divergence-minimizing discriminator can be seen as approximating the vector field of a gradient flow in the Wasserstein space, induced by a monotonically increasing function of the density ratio. Our results can be seen as a generalization of those considered in MonoFlow, relating both divergence-minimizing, and IPM-based GANs. Liang (2021) optimize IPM-based generator and discriminator networks jointly, and show that additional regularization on the space of the generator functions is necessary in IPM GANs to attain the optimum. Franceschi et al. (2022) propose **NTK-GANs**, a unifying theory for the optimality of GANs considering neural-network discriminators, and show that the generator in and GAN can be seen as minimizing a cost related to the NTK associated with an infinite-width discriminator.

Normalizing Flows: Popularized by Rezende & Mohamed (2015), normalizing flows leverage the *change-of-variables* formula to learn a transformation from a parametric prior distribution to a target. The network architecture is constrained so as to facilitate easy computation of the Jacobian (Dinh et al., 2015, 2017; Kingma & Dhariwal, 2018). Recent approaches design flows based on autoregressive models (Kingma et al., 2016; Papamakarios et al., 2017; Su & Wu, 2018), or architectures motivated by the Sobolev GAN loss (Mroueh et al., 2019; Mroueh & Rigotti, 2020). Glaser et al. (2021); Ansari et al. (2021) use KL-flow to iteratively improve the noise vector input to GANs.

In the GAN context, consider the generator push-forward distribution $p_g = G_{\#}(p_z)$. For the main results of this paper, **we assume** $G\colon \mathbb{R}^n \to \mathbb{R}^n$, where the generator is a *diffeomorphism* with a well-defined inverse G^{-1}, both G and its inverse being differentiable. Therefore, $z \in \mathbb{R}^n$ is no longer the *latent* representation. Then, by the change-of-variables formula, we have:

$$p_g(\boldsymbol{x}) = p_z(\boldsymbol{z})\, |\det\, \mathrm{J}_G(\boldsymbol{z})|^{-1},\ \text{where}\ \boldsymbol{z} = G^{-1}(\boldsymbol{x}), \tag{4}$$

where in turn, $\mathrm{J}_G(\boldsymbol{z})$ is the Jacobian of the generator. Standard mathematical notations used in this paper, and relevant background on the *Fundamental Lemma of the Calculus of Variations* (Gel'fand & Fomin, 1964) are provided in Appendix A. We now present results discussing the optimal generator in divergence minimizing GANs.

C Optimality of Divergence-minimizing GANs

We now present the proofs for the optimality conditions derived for the divergence minimizing GANs. We also consider an extension, pertaining to a GAN variant that does not directly correspond to the f-divergence, the LSGAN formulation with arbitrarily chosen class labels.

C.1 Optimality of f-GAN (Proof of Theorem 2.1)

We now derive the optimality condition for f-GANs in general, and subsequently analyze each variant considered by Nowozin et al. (2016) (cf. Table 1). Recall the f-GAN optimization:

$$\mathcal{L}_D^f(D;\, G_{t-1}) = -\mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_d}[T(\boldsymbol{x})] + \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_{t-1}}[f^c(T(\boldsymbol{x}))]$$
$$\mathcal{L}_G^f(G; D_t^*, G_{t-1}) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_d}[T^*(\boldsymbol{x})] - \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_{t-1}}[f^c(T^*(\boldsymbol{x}))]\,,$$

where $T(\cdot) = g(D(\cdot))$ is the output of the discriminator $D \in \mathbb{R}$, restricted to a desired domain by mean of an activation function $g(\cdot)$, f^c denotes the Fenchel conjugate of f, and $T^*(\cdot) = g(D^*(\cdot))$. The discriminator optimization in f-GANs has been well studied by Nowozin et al. (2016); Asokan & Seelamantula (2023) and Yi et al. (2023). For completeness, we summarize the result here. Consider the integral form of the discriminator cost:

$$\mathcal{L}_D^f(D;\, G_{t-1}) = \int_{\mathcal{X}} \underbrace{f^c(T(\boldsymbol{x}))\, p_{t-1}(\boldsymbol{x}) - T(\boldsymbol{x})\, p_d(\boldsymbol{x})}_{\mathcal{F}D}\, \mathrm{d}\boldsymbol{x}$$

The integrand $\mathcal{F}$ can be optimized pointwise with respect to T, to derive the optimality condition for the discriminator:

$$\left.\frac{\partial f^c(T)}{\partial T}\right|_{T=T^*(\boldsymbol{x})} = \frac{p_d(\boldsymbol{x})}{p_{t-1}(\boldsymbol{x})} = r_{t-1}(\boldsymbol{x}), \qquad \text{where} \qquad T^*(\boldsymbol{x}) = g(D^*(\boldsymbol{x})). \tag{5}$$

The above can be solved for various choices of $g(\cdot)$ and $f^c(\cdot)$, giving rise to the optimal discriminator $D_t^*(\boldsymbol{x})$ in f-GANs. For convenience, we recall the results in Table 1 of the Appendix. Since the optimal discriminator is always a function of the logarithm of the density ratio, we denote the solution as $D_t^*(\ln(r_{t-1}))$.

Consider the f-GAN generator optimization, given D_t^*. Only the term involving the generator samples affects the optimization. The integral form of the loss is given by:

$$\mathcal{L}_G^f(G) = \int_{\mathcal{Z}} f^c(T^*(G(\boldsymbol{z})))\, p_z(\boldsymbol{z})\, \mathrm{d}\boldsymbol{z}.$$

Let the optimal solution be denoted by

$$G_t^*(\boldsymbol{z}) = [G_t^{1^*}(\boldsymbol{z}), G_t^{2^*}(\boldsymbol{z}),\ \ldots,\ G_t^{i^*}(\boldsymbol{z}),\ \ldots,\ G_t^{n^*}(\boldsymbol{z})]^{\mathrm{T}},$$

where $G_t^{i^*}$ denotes the optimum along the i^{th} dimension. Let $\mathcal{L}_{G,i,\epsilon}$ be the loss considering an epsilon perturbation of the i^{th} entry about the optimum, given by:

$$G_{t,i,\epsilon}^*(\boldsymbol{z}) = [G_t^{1^*}(\boldsymbol{z}), G_t^{2^*}(\boldsymbol{z}),\ \ldots,\ G_t^{i^*}(\boldsymbol{z}) + \epsilon\eta(\boldsymbol{z}),\ \ldots,\ G_t^{n^*}(\boldsymbol{z})]^{\mathrm{T}},$$

where $\eta(\boldsymbol{z})$ is drawn from a family of compactly supported, infinitely differentiable functions. The loss can now be written as a function of ϵ as:

$$\mathcal{L}_{G,i,\epsilon}^f(\epsilon) = \int_{\mathcal{Z}} f^c(T^*(G_{t,i,\epsilon}^*(\boldsymbol{z})))\, p_z(\boldsymbol{z})\, \mathrm{d}\boldsymbol{z}.$$

Leveraging the chain rule and computing the derivative with respect to ϵ yields:

$$\left.\frac{\partial \mathcal{L}_{G,i,\epsilon}^f(\epsilon)}{\partial \epsilon}\right|_{\epsilon=0} = \int_{\mathcal{Z}} \left.\frac{\partial f^c}{\partial T}\right|_{T=T^*(G_{t,i,\epsilon}^*)} \left.\frac{\partial T^*}{\partial D}\right|_{D=D_t^*(\ln r_{t-1})} \cdot \left.\frac{\partial D_t^*}{\partial y}\right|_{y=\ln r_{t-1}(G_{t,i,\epsilon}^*)} \left.\frac{\partial}{\partial \epsilon}\left(\ln\big(r_{t-1}(G_{t,i,\epsilon}^*)(\boldsymbol{z})\big)\right)\, p_z(\boldsymbol{z})\right|_{\epsilon=0} \mathrm{d}\boldsymbol{z}.$$

Table 1: Various f-GANs (Nowozin et al., 2016), given the activation function g and the Fenchel conjugate f^c. The corresponding optimal discriminator $D_t^*(\boldsymbol{x})$, derived via pointwise optimization, and the corresponding coefficient function $\mathscr{C}$.

f-divergence	$g(D)$	$f^c(T)$	$D_t^*(\ln(r_{t-1}))$	$\mathscr{C}(\boldsymbol{x};\, p_d,\, p_{t-1})$
Kullback-Leibler (KL)	D	e^{T-1}	$1+\ln\left(r_{t-1}(\boldsymbol{x})\right)$	$r_{t-1}(\boldsymbol{x})$
Reverse KL	$-e^{-D}$	$-1-\ln(-T)$	$\ln\left(r_{t-1}(\boldsymbol{x})\right)$	1
Pearson-χ^2	D	$\frac{1}{4}T^2+T$	$2\left(r_{t-1}(\boldsymbol{x})-1\right)$	$2r_{t-1}^2(\boldsymbol{x})$
Squared-Hellinger	$1-e^{-D}$	$\frac{T}{1-T}$	$\frac{1}{2}\ln\left(r_{t-1}(\boldsymbol{x})\right)$	$\frac{1}{2}\sqrt{r_{t-1}(\boldsymbol{x})}$
SGAN	$-\ln\left(1+e^{-D}\right)$	$-\ln\left(1-e^{T}\right)$	$\ln\left(r_{t-1}(\boldsymbol{x})\right)$	$r_{t-1}^2(\boldsymbol{x})\left(r_{t-1}(\boldsymbol{x})+1\right)^{-1}$

From the optimality condition given in Equation (5), and relations in Table 1, we have:

$$\begin{aligned}
\left.\frac{\partial \mathcal{L}^f_{G,i,\epsilon}(\epsilon)}{\partial \epsilon}\right|_{\epsilon=0} &= \int_{\mathcal{Z}} r_{t-1}(\boldsymbol{x}) g'\left(D_t^*(\ln r_{t-1}(\boldsymbol{x}))\right) \left.\frac{\partial D_t^*}{\partial y}\right|_{y=\ln r_{t-1}} \cdot \\
&\qquad \left.\frac{\partial}{\partial x_i}\left(\ln(r_{t-1}(\boldsymbol{x}))\right)\right|_{\boldsymbol{x}=G_t^*(\boldsymbol{z})} \frac{\partial G^*_{t,i,\epsilon}}{\partial \epsilon}\, p_z(\boldsymbol{z})\, \mathrm{d}\boldsymbol{z} \\
&= \int_{\mathcal{Z}} \underbrace{r_{t-1}(\boldsymbol{x}) g'\left(D_t^*(\ln r_{t-1}(\boldsymbol{x}))\right) D_t^{*\prime}(y)\big|_{y=\ln r_{t-1}}}_{\mathscr{C}(\boldsymbol{x};\, p_d,\, p_{t-1})} \cdot \\
&\qquad \left.\frac{\partial}{\partial x_i}\left(\ln(r_{t-1}(\boldsymbol{x}))\right)\right|_{\boldsymbol{x}=G_t^*(\boldsymbol{z})} \eta(\boldsymbol{z})\, p_z(\boldsymbol{z})\, \mathrm{d}\boldsymbol{z} = 0,
\end{aligned}$$

where $g'(t)$ denotes the derivatives of the activation function with respect to D evaluated at D_t^*, and $D_t^{*\prime}(y)$ denotes the derivative of the optimal discriminator function with respect to $y = \ln(r_{t-1}(\boldsymbol{x}))$, evaluated at $\ln(r_{t-1}(\boldsymbol{x}))$. As in the case of SGANs, from the *Fundamental Lemma of Calculus of Variations*, we have:

$$\mathscr{C}(\boldsymbol{x};\, p_d,\, p_{t-1}) \frac{\partial}{\partial x_i}\left(\ln(r_{t-1}(\boldsymbol{x}))\right)\Big|_{\boldsymbol{x}=G_t^*(\boldsymbol{z})} = 0, \qquad \forall\, \boldsymbol{z} \in \mathcal{Z}.$$

The above can be vectorized over all i, giving rise to the result provided in Theorem 2.1:

$$\mathscr{C}(\boldsymbol{x};\, p_d,\, p_{t-1}) \nabla_{\boldsymbol{x}} \left(\ln r_{t-1}(\boldsymbol{x})\right)\Big|_{\boldsymbol{x}=G_t^*(\boldsymbol{z})} = \mathbf{0}, \qquad \forall\, \boldsymbol{z} \in \mathcal{Z}.$$

The coefficient $\mathscr{C}(\boldsymbol{x};\, p_d,\, p_{t-1})$ can be derived for each f-GAN (given in Table 1), we discuss the results here.

KL divergence: Consider the f-GAN with the Kullback-Leibler divergence. We have $g'(D_t^*) = 1$ and $D_t^{*\prime}(\boldsymbol{x}) = 1$, which gives us $\mathscr{C}(\boldsymbol{x};\, p_d,\, p_{t-1}) = r_{t-1}(\boldsymbol{x})$. Recall that the density ratio is given by $r_{t-1}(G_t^*(\boldsymbol{z})) = \frac{p_d(G_t^*(\boldsymbol{z}))}{p_{t-1}(G_t^*(\boldsymbol{z}))}$. Since p_{t-1} denotes the push-forward distribution of the generator of the previous iteration, for sufficiently small learning rates, the generator samples at time t are sufficiently close to those at $t-1$, and $p_{t-1}(G_t^*(\boldsymbol{z}))$ is non-zero. However, if the generated samples are far from the data density, $p_d(G_t^*(\boldsymbol{z}))$ is close to zero, resulting in vanishing gradients while training — Even if the scores do not match, the training loss is zero, as $p_d(G_{\theta_t}(\boldsymbol{z})) \to 0$.

Reverse-KL (RKL) divergence: For the reverse-KL-based f-GAN, we have $g'(D_t^*) = r_{t-1}^{-1}(\boldsymbol{x})$ and $D_t^{*\prime}(\boldsymbol{x}) = 1$. As a consequence, the coefficient $\mathscr{C}(\boldsymbol{x};\, p_d,\, p_{t-1})$ is unity. Therefore, when trained with the RKL loss, it is clear that the generator would not suffer from vanishing gradients. This observation is consistent with the literature, as Nguyen et al. (2017) and Shannon et al. (2020) have both observed that the non-saturating GAN loss can be seen as a *smoothened* RKL loss.

Pearson-χ^2 divergence: The Pearson-χ^2 GAN can be seen as a special case of LSGAN. Here, we have $g'(D_t^*) = 1$ and $D_t^{*\prime}(\boldsymbol{x}) = r_{t-1}(\boldsymbol{x})$. The coefficient $\mathscr{C}(\boldsymbol{x};\, p_d,\, p_{t-1}) = r_{t-1}^2(\boldsymbol{x})$ grows quadratically in p_d, resulting in vanishing gradients in a more pronounced manner than in KL-GANs. We discuss the effect of choosing alternative class labels in LSGAN, those that do not lead to the Pearson-χ^2 GAN, in Appendix C.3.

Squared-Hellinger divergence: In the Squared-Hellinger GAN formulation, we have $g'(D_t^*) = r_{t-1}^{-\frac{1}{2}}(\boldsymbol{x})$ and $D_t^{*\prime}(\boldsymbol{x}) = \frac{1}{2}$, which yields $\mathscr{C}(\boldsymbol{x};\, p_d,\, p_{t-1}) = \frac{1}{2} r_{t-1}^{\frac{1}{2}}(\boldsymbol{x})$. As the coefficient only decays as the square-root of p_d, we expect that the Squared-Hellinger GAN is relatively more stable, compared to the Pearson-χ^2-divergence based counterpart. This was observed empirically by Nowozin et al. (2016).

C.2 Optimality of SGAN

As a special case, we consider the original SGAN optimization proposed by Goodfellow et al. (2014):

$$\min_G \max_D \left\{ \mathbb{E}_{\boldsymbol{x}\sim p_d}[\ln D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_z}[\ln(1 - D(G(\boldsymbol{z})))] \right\}.$$

In practice, the optimization is an alternating one, wherein the discriminator D_t is derived given the generator of the previous iteration, G_{t-1}, and the subsequent generator optimization involves computing G_t, given D_t and G_{t-1}. For simplicity, we denote the push-forward distribution at iteration t as $p_t(\boldsymbol{x}) = G_{t,\#}(p_z(\boldsymbol{z}))$. Within this formulation, the optimization becomes:

$$\mathcal{L}_D^S(D; G_{t-1}) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_d}[\ln D(\boldsymbol{x})] + \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_{t-1}}[\ln(1 - D(\boldsymbol{x}))], \text{ where } D_t^*(\boldsymbol{x}) = \arg\max_D \left\{ \mathcal{L}_D^S \right\} \quad (6)$$

$$\text{and } \mathcal{L}_G^S(G; D_t^*, G_{t-1}) = \mathop{\mathbb{E}}_{\boldsymbol{z}\sim p_z}[\ln(1 - D_t^*(G(\boldsymbol{z})))], \text{ where } G_t^*(\boldsymbol{x}) = \arg\min_G \left\{ \mathcal{L}_G^S \right\}. \quad (7)$$

Ideally, both the discriminator and generator would converge to the optimal solution at $t = 1$. However, in practice, through a stochastic-gradient-descent update, under mild assumptions, the alternating optimization converges to the desired optimum (Franceschi et al., 2022), *i.e.,* p_g converges to the p_d in the limit ($\lim_{t\to\infty} p_t(\boldsymbol{x}) = p_d(\boldsymbol{x})$). Optimization of the loss in Equation (6) can be carried out pointwise (Goodfellow et al., 2014), with the resulting optimum given by:

$$D_t^*(\boldsymbol{x}) = \frac{p_d(\boldsymbol{x})}{p_d(\boldsymbol{x}) + p_{t-1}(\boldsymbol{x})}. \quad (8)$$

Theorem C.1. *Consider the generator cost in Equation* (7)*, and the optimal discriminator given by Equation* (8)*. The* ***optimal SGAN generator*** *that minimizes $\mathcal{L}_G^S$ satisfies*

$$\nabla_{\boldsymbol{x}} \ln\left(p_{t-1}(\boldsymbol{x})\right)\big|_{\boldsymbol{x}=G_t^*(\boldsymbol{z})} = \nabla_{\boldsymbol{x}} \ln\left(p_d(\boldsymbol{x})\right)\big|_{\boldsymbol{x}=G_t^*(\boldsymbol{z})}, \quad (9)$$

where $\boldsymbol{z} \sim p_z$, and $\nabla_{\boldsymbol{x}} \ln p_{t-1}$ is the score of the push-forward generator distribution $G_{t-1,\#}(p_z)$.

Proof. Recall the SGAN generator optimization problem:

$$\mathcal{L}_G^S(G;\, D_t^*, G_{t-1}) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_d}[\ln(D_t^*(\boldsymbol{x}))] + \mathop{\mathbb{E}}_{\boldsymbol{z}\sim p_z}[\ln(1 - D_t^*(G(\boldsymbol{z})))]$$

$$G_t^*(\boldsymbol{x}) = \arg\min_G \left\{ \mathcal{L}_G^S(G;\, D_t^*, G_{t-1}) \right\}, \quad \text{where} \quad D_t^*(\boldsymbol{x}) = \frac{p_d(\boldsymbol{x})}{p_{t-1}(\boldsymbol{x}) + p_d(\boldsymbol{x})}.$$

The optimal discriminator was originally derived through a point-wise optimization by Goodfellow et al. (2014), but later shown to be consistent with the functional form of optimization by Asokan & Seelamantula (2023). Since the expectation with respect to the data term in $\mathcal{L}_G^S$ does not involve the generator samples at the current iteration t, it can be ignored with respect to the optimization. A similar approach was considered by Franceschi et al. (2022) in analyzing the NTK-GAN formulation. Expanding the integral, and substitute in for the optimal discriminator D_t^* yields:

$$\mathcal{L}_G^S(G) = \int_{\mathcal{Z}} \ln\left(\frac{p_{t-1}(G(\boldsymbol{z}))}{p_{t-1}(G(\boldsymbol{z})) + p_d(G(\boldsymbol{z}))}\right) p_z(\boldsymbol{z})\, \mathrm{d}\boldsymbol{z},$$

where $\mathcal{Z}$ denotes the support of the input distribution p_z. The optimization of $\mathcal{L}_G^S$ is a functional one, and can be found by computing the first variation, and setting it to zero under the *Fundamental*

Lemma of Calculus of Variations. The perturbed loss $\mathcal{L}^S_{G,i,\epsilon}$ is defined as in the case of f-GANs (cf. Appendix C.1):

$$\mathcal{L}^S_{G,i,\epsilon}(\epsilon) = \int_{\mathcal{Z}} \ln\left(\frac{p_{t-1}\left(G^*_{t,i,\epsilon}(\boldsymbol{z})\right)}{p_{t-1}\left(G^*_{t,i,\epsilon}(\boldsymbol{z})\right) + p_d\left(G^*_{t,i,\epsilon}(\boldsymbol{z})\right)}\right) p_z(\boldsymbol{z})\,\mathrm{d}\boldsymbol{z},$$

Differentiating $\mathcal{L}_{G,i,\epsilon}$ with respect to epsilon and equating it to zero at $\epsilon = 0$ yields:

$$\begin{aligned}\frac{\partial \mathcal{L}^S_{G,i,\epsilon}(\epsilon)}{\partial \epsilon}\bigg|_{\epsilon=0} &= \int_{\mathcal{Z}} \left(\frac{p_{t-1}\left(G^*_{t,i,\epsilon}(\boldsymbol{z})\right) + p_d\left(G^*_{t,i,\epsilon}(\boldsymbol{z})\right)}{p_{t-1}\left(G^*_{t,i,\epsilon}(\boldsymbol{z})\right)}\right)\bigg|_{\epsilon=0} \cdot \\ &\qquad \underbrace{\frac{\partial}{\partial \epsilon}\left(\frac{p_{t-1}\left(G^*_{t,i,\epsilon}(\boldsymbol{z})\right)}{p_{t-1}\left(G^*_{t,i,\epsilon}(\boldsymbol{z})\right) + p_d\left(G^*_{t,i,\epsilon}(\boldsymbol{z})\right)}\right)}_{\mathrm{T}_1} p_z(\boldsymbol{z})\,\mathrm{d}\boldsymbol{z} \\ &= 0.\end{aligned}$$

Let $\boldsymbol{x} = G^*_{t,i,\epsilon}(\boldsymbol{z})\big|_{\epsilon=0} = G^*_t(\boldsymbol{z})$. The term T_1 can be simplified as:

$$\begin{aligned}\mathrm{T}_1 &= \left(\frac{p_d(G^*_{t,i,\epsilon}(\boldsymbol{z}))\frac{\partial p_{t-1}(\boldsymbol{y})}{\partial y_i}\big|_{\boldsymbol{y}=G^*_{t,i,\epsilon}(\boldsymbol{z})} - p_{t-1}(G^*_{t,i,\epsilon}(\boldsymbol{z}))\frac{\partial p_d(\boldsymbol{y})}{\partial y_i}\big|_{\boldsymbol{y}=G^*_{t,i,\epsilon}(\boldsymbol{z})}}{\left(p_d(G^*_{t,i,\epsilon}(\boldsymbol{z})) + p_{t-1}(G^*_{t,i,\epsilon}(\boldsymbol{z}))\right)^2}\right)\bigg|_{\epsilon=0}\eta(\boldsymbol{z}) \\ &= \left(\frac{p_d(\boldsymbol{x})\frac{\partial p_{t-1}(\boldsymbol{x})}{\partial x_i} - p_{t-1}(\boldsymbol{x})\frac{\partial p_d(\boldsymbol{x})}{\partial x_i}}{\left(p_d(\boldsymbol{x}) + p_{t-1}(\boldsymbol{x})\right)^2}\right)\eta(\boldsymbol{z}).\end{aligned}$$

Substituting back for T_1 in $\frac{\partial \mathcal{L}^S_{G,i,\epsilon}(\epsilon)}{\partial \epsilon}$ and simplifying yields:

$$\frac{\partial \mathcal{L}^S_{G,i,\epsilon}(\epsilon)}{\partial \epsilon}\bigg|_{\epsilon=0} = \int_{\mathcal{Z}} p_z(\boldsymbol{z})\left(\frac{p_d(\boldsymbol{x})p'_{t-1,i}(\boldsymbol{x}) - p_{t-1}(\boldsymbol{x})p'_{d,i}(\boldsymbol{x})}{p_{t-1}(\boldsymbol{x})\left(p_d(\boldsymbol{x}) + p_{t-1}(\boldsymbol{x})\right)}\right)\eta(\boldsymbol{z})\bigg|_{\boldsymbol{x}=G^*_t(\boldsymbol{z})} = 0,$$

where $p'_{t-1,i}$ and $p'_{d,i}$ denote the derivative of p_{t-1} and p_d, respectively, with respect to x_i, the i^{th} entry of the argument $\boldsymbol{x}$. Then, from the *Fundamental Lemma of the Calculus of Variations*, we have:

$$p_z(\boldsymbol{z})\left(\frac{p_d(\boldsymbol{x})p'_{t-1,i}(\boldsymbol{x}) - p_{t-1}(\boldsymbol{x})p'_{d,i}(\boldsymbol{x})}{p_{t-1}(\boldsymbol{x})\left(p_d(\boldsymbol{x}) + p_{t-1}(\boldsymbol{x})\right)}\right)\bigg|_{\boldsymbol{x}=G^*_t(\boldsymbol{z})} = 0, \qquad \forall\ \boldsymbol{z} \in \mathcal{Z}.$$

Since $p_z(\boldsymbol{z})$ is non-zero over its support $\mathcal{Z}$, and if $p_{t-1}(\boldsymbol{x}) \neq 0$ for all $\boldsymbol{x} = G^*_t(\boldsymbol{z})$ (which is a reasonable assumption to make, since p_{t-1} is the push-forward distribution of the generator), the optimality condition becomes:

$$p_d(\boldsymbol{x})p'_{t-1,i}(\boldsymbol{x}) - p_{t-1}(\boldsymbol{x})p'_{d,i}(\boldsymbol{x})\Big|_{\boldsymbol{x}=G^*_t(\boldsymbol{z})} = 0, \qquad \forall\ \boldsymbol{z} \in \mathcal{Z}.$$

Rearranging and simplifying yields:

$$\frac{\partial \ln p_{t-1}(\boldsymbol{x})}{\partial x_i}\bigg|_{\boldsymbol{x}=G^*_t(\boldsymbol{z})} = \frac{\partial \ln p_d(\boldsymbol{x})}{\partial x_i}\bigg|_{\boldsymbol{x}=G^*_t(\boldsymbol{z})}, \qquad \forall\ \boldsymbol{z} \in \mathcal{Z}.$$

Since the analysis holds identically for all i, we have:

$$\nabla_{\boldsymbol{x}} \ln\left(p_{t-1}(\boldsymbol{x})\right)\big|_{\boldsymbol{x}=G^*_t(\boldsymbol{z})} = \nabla_{\boldsymbol{x}} \ln\left(p_d(\boldsymbol{x})\right)\big|_{\boldsymbol{x}=G^*_t(\boldsymbol{z})}, \qquad \forall\ \boldsymbol{z} \in \mathcal{Z}.$$

which is the desired result of Theorem C.1. □

The optimality condition on $G^*_t(\boldsymbol{x})$ can be derived element-wise through the Fundamental Lemma of Calculus of Variations and vectorized to yield the result in Theorem C.1. The above result is valid only for those $\boldsymbol{x}$ such that both $p_d(\boldsymbol{x})$ and $p_{t-1}(\boldsymbol{x})$ are non-zero.

C.3 Non-divergence-minimizing GAN Formulations

In this section, we consider an example GAN formulation that does not lie within the divergence minimization framework. The results serve to show that the proposed approach can be applied to any existing GAN variant.

Least-squares GAN: Consider the LSGAN formulation presented by Mao et al. (2017) with the discriminator and generator loss given by:

$$\mathcal{L}_D^{LS}(D;\, G_{t-1}) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_d}\left[(D(\boldsymbol{x})-b)^2\right] + \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_{t-1}}\left[(D(\boldsymbol{x})-a)^2\right] \text{ and}$$
$$\mathcal{L}_G^{LS}(G;\, D_t^*,\, G_{t-1}) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim p_d}\left[(D_t^*(\boldsymbol{x})-c)^2\right] + \mathop{\mathbb{E}}_{\boldsymbol{z}\sim p_z}\left[(D_t^*(G(\boldsymbol{z}))-c)^2\right],$$

respectively, where a and b are the class-labels assigned by the discriminator to real and fake samples, respectively. The generator is trained to create samples such that they are classified as c by the discriminator. The discriminator optimization can be carried out pointwise, giving rise to the optimal discriminator:

$$D_t^*(\boldsymbol{x}) = \frac{a p_{t-1}(\boldsymbol{x}) + b p_d(x)}{p_{t-1}(\boldsymbol{x}) + p_d(x)}. \tag{10}$$

As in the case of SGANs, the generator loss can be expanded into the integral form, and evaluated at perturbed location about the optimal solution $G^*_{t,i,\epsilon}$, which yields:

$$\begin{aligned}
\mathcal{L}^{LS}_{G,i,\epsilon}(\epsilon) &= \int_{\mathcal{Z}} \left(D_t^*(G^*_{t,i,\epsilon}(\boldsymbol{z})) - c\right)^2 p_z(\boldsymbol{z})\, \mathrm{d}\boldsymbol{z} \\
\Rightarrow \left.\frac{\partial \mathcal{L}^{LS}_{G,i,\epsilon}(\epsilon)}{\partial \epsilon}\right|_{\epsilon=0} &= \int_{\mathcal{Z}} 2\left(D_t^*(G^*_{t,i,\epsilon}(\boldsymbol{z})) - c\right) \left.\frac{\partial D_t^*(G^*_{t,i,\epsilon}(\boldsymbol{z}))}{\partial \epsilon}\right|_{\epsilon=0} p_z(\boldsymbol{z})\, \mathrm{d}\boldsymbol{z} \\
&= \int_{\mathcal{Z}} 2\left(D_t^*(\boldsymbol{x}) - c\right) \left.\frac{\partial D_t^*(\boldsymbol{x})}{\partial x_i}\right|_{\boldsymbol{x}=G_t^*(\boldsymbol{z})} \left.\eta(\boldsymbol{z}) p_z(\boldsymbol{z})\right|_{x=G^*_{t,i,\epsilon}(\boldsymbol{z})} \mathrm{d}\boldsymbol{z} = 0
\end{aligned} \tag{11}$$

Given the optimal LSGAN discriminator in Equation (10), we have:

$$\begin{aligned}
\left(D_t^*(\boldsymbol{x}) - c\right) &= \frac{(a-c)p_{t-1}(\boldsymbol{x}) + (b-c)p_d(\boldsymbol{x})}{p_{t-1}(\boldsymbol{x}) + p_d(\boldsymbol{x})} \quad \text{and} \\
\frac{\partial D_t^*(\boldsymbol{x})}{\partial x_i} &= \frac{(b-a)\left(p_d(\boldsymbol{x}) p'_{t-1,i}(\boldsymbol{x}) - p_{t-1}(\boldsymbol{x}) p'_{d,i}(\boldsymbol{x})\right)}{\left(p_{t-1}(\boldsymbol{x}) + p_d(\boldsymbol{x})\right)^2}
\end{aligned}$$

Substituting for the above into Equation (11) yields:

$$\int_{\mathcal{Z}} \underbrace{\frac{(b-a)\left((a-c)p_{t-1}(\boldsymbol{x}) + (b-c)p_d(\boldsymbol{x})\right)}{\left(p_{t-1}(\boldsymbol{x}) + p_d(\boldsymbol{x})\right)^3}}_{\mathcal{C}(\boldsymbol{x};\, p_{t-1}, p_d, a, b, c)} \cdot \left(p_d(\boldsymbol{x}) p'_{t-1,i}(\boldsymbol{x}) - p_{t-1}(\boldsymbol{x}) p'_{d,i}(\boldsymbol{x})\right) \Big|_{x=G^*_{t,i,\epsilon}(\boldsymbol{z})} \eta(\boldsymbol{z}) p_z(\boldsymbol{z})\, \mathrm{d}\boldsymbol{z} = 0,$$

where $\mathcal{C}(\boldsymbol{x};\, p_{t-1}, p_d, a, b, c)$ is the coefficient term, similar to the one seen in the f-GAN formulation, that also depends on the choice of class-labels (a, b, c). From the *Fundamental Lemma of Calculus of Variations*, we have:

$$\mathcal{C}(\boldsymbol{x};\, p_{t-1}, p_d, a, b, c)\left(p_d(\boldsymbol{x})\, p'_{t-1,i}(\boldsymbol{x}) - p_{t-1}(\boldsymbol{x})\, p'_{d,i}(\boldsymbol{x})\right)\Big|_{x=G^*_{t,i,\epsilon}(\boldsymbol{z})} = 0,$$

As in the case of f-GANs, we see that the least-squares GAN also results in a score-matching loss, when $\mathcal{C}(\boldsymbol{x};\, p_{t-1}, p_d, a, b, c)$ is non-zero. Mao et al. (2017) propose two choices of class labels – (i) $(a, b, c) = (-1, 0, 1)$, which satisfy the conditions that $b - c = 1$ and $a - c = -1$, resulting in the Pearson-χ^2 divergence-based GANs; and (2) $(a, b, c) = (0, 1, 1)$, which leads to stabler training.

From the solution above, we see that,

$$\text{When } (a,b,c) = (-1,0,1), \text{ we have } \mathcal{C}(\boldsymbol{x};\, p_{t-1}, p_d, a, b, c) = \frac{1}{(p_{t-1}(\boldsymbol{x}) - p_d(\boldsymbol{x}))^2}, \text{ and}$$

$$\text{When } (a,b,c) = (0,1,1), \text{ we have } \mathcal{C}(\boldsymbol{x};\, p_{t-1}, p_d, a, b, c) = \frac{-p_{t-1}(\boldsymbol{x})}{(p_{t-1}(\boldsymbol{x}) - p_d(\boldsymbol{x}))^3}.$$

For either case, for sufficiently small learning rates, the updated sample $\boldsymbol{x} = G_t^*(\boldsymbol{z})$ is sufficiently close to the sample generated at the previous iteration, and we have $p_{t-1}(\boldsymbol{x}) > 0$. As a result, we see that even when the loss does not correspond to a divergence minimizing cost, the class label (a, b, c) can be chosen such that the LSGAN generator optimization results in a score-matching cost.

C.4 Computing the Score of the Generator (Proof of Lemma 2.2)

Consider the push-forward generator distribution at time t, given by $p_t(\boldsymbol{x}) = G_{\theta_t,\#}(p_z)$, where $p_z = \mathcal{N}(\boldsymbol{z}; \mu_z, \Sigma_z)$. We assume that the generator $G_{\theta_t} : \mathbb{R}^n \to \mathbb{R}^n$ is an invertible function, with the inverse given by $G_{\theta_t}^{-1}$. Then, by the change-of-variables formula, we have:

$$p_t(\boldsymbol{x}) = p_z(G_{\theta_t}^{-1}(\boldsymbol{x}))\ \left|\det \mathrm{J}_{G_{\theta_t}^{-1}}(\boldsymbol{x})\right|.$$

If the generator is invertible, we have,

$$p_t(\boldsymbol{x}) = p_z(G_{\theta_t}^{-1}(\boldsymbol{x}))\ \left|\det \mathrm{J}^{-1}_{G_{\theta_t}}(G_{\theta_t}^{-1}(\boldsymbol{x}))\right| = p_z(G_{\theta_t}^{-1}(\boldsymbol{x}))\ \left|\det \mathrm{J}_{G_{\theta_t}}(G_{\theta_t}^{-1}(\boldsymbol{x}))\right|^{-1}.$$

Then, the score of the generator is given by:

$$\begin{aligned}\nabla_{\boldsymbol{x}} \ln\left(p_t(\boldsymbol{x})\right) &= \nabla_{\boldsymbol{x}} \ln\left(p_z(G_{\theta_t}^{-1}(\boldsymbol{x}))\ \left|\det \mathrm{J}_{G_{\theta_t}}(G_{\theta_t}^{-1}(\boldsymbol{x}))\right|^{-1}\right)\\ &= \nabla_{\boldsymbol{x}}\left(\ln\left(p_z(G_{\theta_t}^{-1}(\boldsymbol{x}))\right) - \ln\left|\det \mathrm{J}_{G_{\theta_t}}(G_{\theta_t}^{-1}(\boldsymbol{x}))\right|\right)\end{aligned}$$

Then, given the transformation $\boldsymbol{x} = G_{\theta_t}(\boldsymbol{z})$, we have

$$\nabla_{\boldsymbol{x}} \ln\left(p_t(\boldsymbol{x})\right) = \mathrm{J}^{-\mathrm{T}}_{G_{\theta_t}}(\boldsymbol{z})\left(\nabla_{\boldsymbol{z}}\left(\ln\left(p_z(\boldsymbol{z})\right) - \ln\left|\det \mathrm{J}_{G_{\theta_t}}(\boldsymbol{z})\right|\right)\right)$$

In most GAN frameworks, p_z is set to be the standard Gaussian $\mathcal{N}(\boldsymbol{z};\, \mathbf{0}, \mathbb{I}$. Simplifying for the score of the Gaussian, we get

$$\nabla_{\boldsymbol{x}} \ln\left(p_t(\boldsymbol{x})\right) = \mathrm{J}^{-\mathrm{T}}_{G_{\theta_t}}(\boldsymbol{z})\Bigg(-\boldsymbol{z} - \underbrace{\nabla_{\boldsymbol{z}} \ln\left|\det \mathrm{J}_{G_{\theta_t}}(\boldsymbol{z})\right|}_{\mathrm{T}_1}\Bigg),$$

which is the desired result of Lemma 2.2. In practice, the Jacobian of the generator can be computed using automatic differentiation in standard libraries such as TensorFlow (Abadi et al., 2016) or PyTorch (Paszke et al., 2019). The term T_1 can further be simplified though well-known matrix differentiation properties. Consider the following:

$$\begin{aligned}\mathrm{T}_1 &= \nabla_{\boldsymbol{z}} \ln\left|\det \mathrm{J}_{G_{\theta_t}}(\boldsymbol{z})\right|\\ &= \nabla_{\boldsymbol{z}}\mathrm{J}_{G_{\theta_t}} \otimes \nabla_{\mathrm{J}_{G_{\theta_t}}} \ln\left|\det \mathrm{J}_{G_{\theta_t}}(\boldsymbol{z})\right|,\end{aligned}$$

where $\nabla_{\boldsymbol{z}}\mathrm{J}_{G_{\theta_t}}$ denotes a Hessian tensor in $\mathbb{R}^{n\times n\times d}$, with the $(i,j,k)^{th}$ entry given by $\left[\nabla_{\boldsymbol{z}}\mathrm{J}_{G_{\theta_t}}\right]_{i,j,k} = \frac{\partial[\mathrm{J}_{G_{\theta_t}}]_{i,j}}{\partial z_k}$. Applying the matrix identity $\nabla_{\boldsymbol{M}} \ln|\det \boldsymbol{M}| = \boldsymbol{M}^{-\mathrm{T}}$ (Petersen et al., 2008) yields:

$$\begin{aligned}\mathrm{T}_1 &= \left(\nabla_{\boldsymbol{z}}\mathrm{J}_{G_{\theta_t}} \otimes \mathrm{J}^{-\mathrm{T}}_{G_{\theta_t}}\right)(\boldsymbol{z}),\\ \text{with entries given by}\quad [\mathrm{T}_1]_i &= \sum_{j,k}\left[\nabla_{\boldsymbol{z}}\mathrm{J}_{G_{\theta_t}}\right]_{i,j,k} \cdot \left[\mathrm{J}^{-\mathrm{T}}_{G_{\theta_t}}\right]_{j,k};\quad i = 1,2,\ldots,d,\end{aligned}$$

where the Hessian tensor can be computed either through automatic differential, or approximated by the Jacobian outer product.

C.5 ScoreGANs with Rectangular Jacobian Matrices

We now extend the results of Appendix C.4 to the scenario when $G_{\theta_t} : \mathbb{R}^d \to \mathbb{R}^n;\ d \ll n$. Papamakarios et al. (2021) showed that when the data $\boldsymbol{x} \in \mathbb{R}^n$ is assumed to lie in a low, d-dimensional manifold by means of the mapping (G_{θ_t}), we can define the metric $\mathrm{M}(\boldsymbol{z})$ induced on the space $\mathcal{X}$ as:

$$\mathrm{M}(\boldsymbol{z}) = \mathrm{J}^{\mathrm{T}}_{G_{\theta_t}}(\boldsymbol{z})\mathrm{J}_{G_{\theta_t}}(\boldsymbol{z}).$$

Then, the change-of-variables formula for the transformation of random variables with measures defined over $\mathcal{X}$ is:

$$p_t(\boldsymbol{x}) = p_z(G^{-1}_{\theta_t}(\boldsymbol{x}))\ \left(\det \mathrm{M}(G^{-1}_{\theta_t}(\boldsymbol{x}))\right)^{-\frac{1}{2}}.$$

An analysis similar to the one provided in Appendix C.4 can now be applied to derive the following approximation:

$$\nabla_{\boldsymbol{x}} \ln\left(p_t(\boldsymbol{x})\right) \approx \mathrm{J}^{\dagger^{\mathrm{T}}}_{G_{\theta_t}}(\boldsymbol{z})\Bigg(-z - \underbrace{\frac{1}{2}\nabla_{\boldsymbol{z}} \ln\det\left(\mathrm{J}^{\mathrm{T}}_{G_{\theta_t}}\mathrm{J}_{G_{\theta_t}}\right)}_{\mathrm{T}_1}\Bigg),$$

where $\mathrm{J}^{\dagger}_{G_{\theta_t}}$ denotes the pseudoinverse of the Jacobian matrix. Further, simplifying T_1 using the standard matrix identity $\nabla_{\boldsymbol{A}} \ln|\det \boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}| = 2\boldsymbol{A}^{\dagger^{\mathrm{T}}}$ (Petersen et al., 2008) yields

$$\begin{aligned} \mathrm{T}_1 &= \nabla_{\boldsymbol{z}}\mathrm{J}_{G_{\theta_t}}(\boldsymbol{z}) \otimes \mathrm{J}^{\dagger^{\mathrm{T}}}_{G_{\theta_t}}(\boldsymbol{z}), \\ \text{with entries given by} \quad [\mathrm{T}_1]_i &= \sum_{j,k}\left[\nabla_{\boldsymbol{z}}\mathrm{J}_{G_{\theta_t}}\right]_{i,j,k} \cdot \left[\mathrm{J}^{\dagger^{\mathrm{T}}}_{G_{\theta_t}}\right]_{j,k}; \quad i = 1, 2, \ldots, d. \end{aligned}$$

While the above result provides a closed-form approximation to the generator density in the most general sense, additional constrained can be enforced on the generator network architecture, as in the case of normalizing flows (Papamakarios et al., 2021) to further simplify computation.

D Additional Experimentation on ScoreGANs

In this appendix, we present additional results from training ScoreGAN on Gaussian data. The training procedure for ScoreGANs is presented in Algorithm 1.

D.1 Additional Experimental Results on Gaussian Learning

Training Parameters: All models are trained using the TensorFlow (Abadi et al., 2016) library. On the unimodal Gaussian experiments, the generator is a linear transformation $\boldsymbol{x} = \boldsymbol{A}\boldsymbol{z} + \boldsymbol{b}$. The target Gaussian is $p_d = \mathcal{N}(5\mathbf{1}_2, 0.75\mathbb{I}_2)$ in 2-D and $p_d = \mathcal{N}(0.7\mathbf{1}_n, 0.02\mathbb{I}_n)$ in the n-D case for $n > 2$. In baseline GAN variants with a network-based discriminator, we use a four-layer perceptron architecture, with 128, 32, 16, and 1 node(s), respectively in each layer. The Leaky-ReLU activation is used across all layers. The networks are trained with the Adam (Kingma & Ba, 2015) optimizer. A batch size of 500 is used. The models are compared using the Wasserstein-2 distance between the target and source Gaussians $\mathcal{W}^{2,2}(p_d, p_g) = \|\boldsymbol{\mu}_d - \boldsymbol{\mu}_g\|_2^2 + \mathrm{Trace}\left(\Sigma_d + \Sigma_g - 2\sqrt{\Sigma_d\Sigma_g}\right)$. On the Gaussian-mixture model (GMM) learning tasks, the generator is a three-layer perceptron architecture, with 32, 16, and 2 node(s), respectively in each layer. The input dimensionality is 100 for all the baseline variants. For ScoreGAN, we compare against both a 2-D input (resulting in a square, invertible Jacobian), and a 100-D input (resulting in a rectangular Jacobian matrix).

Additional results on Gaussian and GMM learning: On the GMM learning task, we consider ablation experiments on training ScoreGAN with, and without, the rectangular Jacobian. In the scenario where the input and output dimensions match, ScoreGAN fails to converge, and the network has insufficient capacity to map an unimodal Gaussian to a multimodal one. Figure 2 presents the generator and data distributions, superimposed on the gradient field over which the generator is optimized, for various baseline variants, ScoreGAN. In the case of the baselines, this corresponds to the gradient of the discriminator, while in ScoreGAN, it is the score of the target dataset. Table 2 compares the *Batch Compute Time* between generator updates for the baseline GANs and ScoreGAN. ScoreGANs are more compute-intensive due to the need for computing the score of the generator network in each update step.

Table 2: A comparison of baseline GAN variants and ScoreGAN in terms of their training time (measured in seconds per batch) on Gaussian learning tasks. ScoreGANs are more compute-intensive due to the need for computing the score of the generator network in each update step.

GAN Variant	Batch Compute Time (seconds/batch)	
	2-D data	128-D data
	Batch size 500	Batch size 100
SGAN	0.4651 ± 0.023	0.2031 ± 0.023
LSGAN	0.4622 ± 0.021	0.1973 ± 0.031
LS-DRAGAN	0.4854 ± 0.020	0.2066 ± 0.019
WGAN-GP	0.4553 ± 0.031	0.1849 ± 0.031
WGAN-R$_d$	0.4427 ± 0.032	0.1932 ± 0.022
Poly-WGAN	0.2316 ± 0.012	0.1571 ± 0.020
GMMN (RBFG)	0.2015 ± 0.020	0.1881 ± 0.031
GMMN (IMQ)	0.1981 ± 0.021	0.1579 ± 0.019
ScoreGAN (Ours)	0.3222 ± 0.022	1.1178 ± 0.015

Algorithm 1: ScoreGAN $-$ Training the GAN generator trained to minimize the distance between its score and the score of the data.

Input: Training data $\boldsymbol{x} \sim p_d$, Gaussian prior distribution $p_z = \mathcal{N}(\mu_{\boldsymbol{z}}, \Sigma_{\boldsymbol{z}})$, Max training iterations T.

Parameters: Batch size M, optimizer learning rate η.

Models: Generator: G_θ; Data score model: $S^d_\phi = \nabla_{\boldsymbol{x}} \ln\left(p_d(\cdot\,;\,\phi)\right)$.

while *t = 1,2,...,T* **do**

- **Sample:** $\boldsymbol{z}_\ell \sim p_z$ – A batch of M noise samples.
- **Sample:** $\boldsymbol{x}_\ell = \mathrm{G}_{\theta_t}(\boldsymbol{z}_\ell)$ – Generator output samples.
- **Compute:** $\mathrm{J}_{G_{\theta_t}}(\boldsymbol{z}_\ell)$ – Jacobian of the generator evaluated at $\boldsymbol{z}_\ell$.
- **Compute:** $\nabla_{\boldsymbol{x}} \ln p_t$ – Score of the generator evaluated at $G_{\theta_t}(\boldsymbol{z}_\ell)$ (cf. Lemma 2.2):
$$\nabla_{\boldsymbol{x}} \ln p_t(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x}_\ell} = -\mathrm{J}^{-\mathrm{T}}_{G_{\theta_t}}\left(\nabla_{\boldsymbol{z}} \ln|\det \mathrm{J}_{G_{\theta_t}}(\boldsymbol{z}_\ell)| + \boldsymbol{z}_\ell\right),$$
- **Compute:** Score-matching-based generator loss (cf. Section 2):
$$\mathcal{L}^{\mathrm{Score}}_G(\theta_t) = \textstyle\sum_{\boldsymbol{x}_\ell} \nabla_{\boldsymbol{x}} \| \ln\left(p_t(\boldsymbol{x}_\ell)\right) - S^d_\phi(\boldsymbol{x}_\ell)\|^2_2.$$
- **Update: Generator** $\mathrm{G}_{\theta_{t+1}} : \theta_{t+1} = \eta \nabla_\theta [\mathcal{L}^{\mathrm{Score}}_G(\theta)]|_{\theta=\theta_t}$ – Generator at θ_{t+1} is the one that minimizes the score matching loss of the generator at θ_t

Output: Samples output by the Generator: $\boldsymbol{x} = \mathrm{G}_{\theta_T}(\boldsymbol{z})$

D.2 Computational Resources

All experiments were carried out using a TensorFlow 2.0 (Abadi et al., 2016) backend. Experiments on NCSN were built atop a publicly available implementation (URL: `https://github.com/Xemnas0/NCSN-TF2.0`). Experiments were performed on SuperMicro workstations with 256 GB of system RAM comprising two NVIDIA GTX 3090 GPUs, each with 24 GB of VRAM.

D.3 Source Code

The TF 2.0 (Abadi et al., 2016) based source code for implementing ScoreGANs is available online at `https://github.com/DarthSid95/ScoreFloWGANs`.

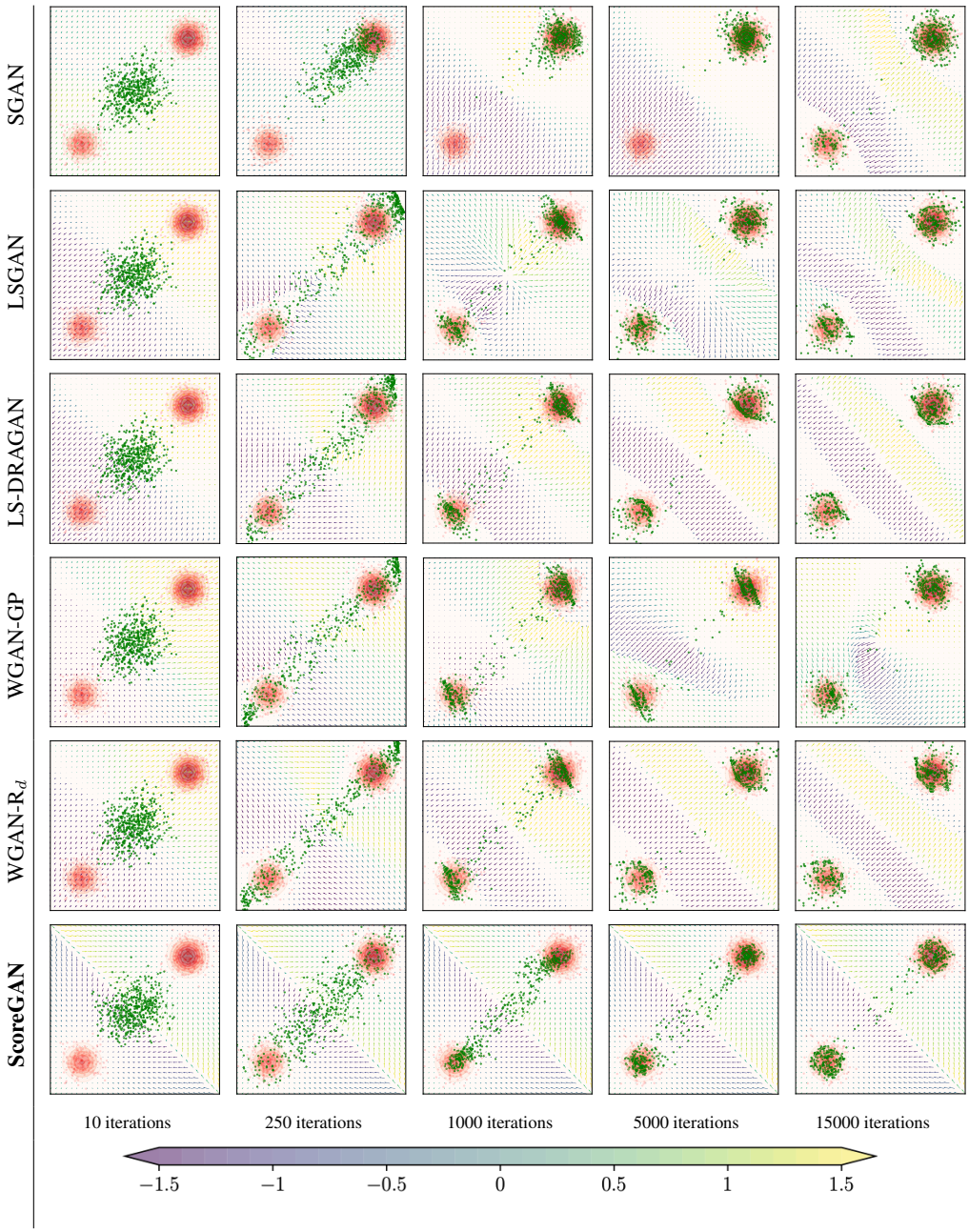


Figure 2: (Color online) Convergence of the generator samples (shown in green) to the target two-component Gaussian (shown in red), $p_d(\boldsymbol{x}) = \frac{1}{5}\mathcal{N}(\boldsymbol{x}; -5\mathbf{1}, \mathbb{I}) + \frac{4}{5}\mathcal{N}(\boldsymbol{x}; 5\mathbf{1}, \mathbb{I})$. The quiver plot depicts the gradient field of the discriminator on baseline variants, and the score of the dataset in the case of ScoreGAN. While SGAN collapses to the more pronounced mode, ScoreGAN converges to both the modes accurately, faster than the baseline counterparts.