Adversarial Variational Optimization of Non-Differentiable Simulators

Gilles Louppe[1] and **Kyle Cranmer**[1]

[1] *New York University*

Complex computer simulators are increasingly used across fields of science to describe generative models tying parameters of an underlying theory to experimental observations. Inference in this setup is often difficult, as simulators rarely provide a way to directly evaluate the likelihood function for a given observation. In this note, we develop a likelihood-free inference algorithm for fitting a forward non-differentiable generative model to (continuous or discrete) observed data. We adapt the adversarial training procedure of generative adversarial networks by replacing the implicit generative network with a domain-based scientific simulator, and solve the resulting non-differentiable minimax problem by minimizing variational upper bounds of the adversarial objectives. Effectively, the procedure results in learning an arbitrarily tight proposal distribution over simulator parameters, such that the corresponding marginal distribution of the generated data matches the observations. [GL: Mention experimental results.] [GL: Add 'so what?' conclusion.]

I. INTRODUCTION

In many fields of science such as particle physics, climatology or population genetics, computer simulators are used to describe complex processes that tie parameters of an underlying theory to high dimensional observations. In most cases, these implicit generative models [12] are specified as procedural implementations of forward stochastic processes that generate discrete or continuous data. Because it is usually computationally intractable, most simulators do not provide a way to directly evaluate the likelihood function for a given observation, thereby making inference difficult. In addition, most scientific simulators are written using opaque low-level programming languages, hence raising the bar for likelihood-free inference algorithms relying e.g. on the simulator being a function with computable derivatives [6] or being a controllable probabilistic program [11].

In this note, we develop a likelihood-free inference algorithm for the point estimation of the parameters of a forward non-differentiable generative model. We adapt the adversarial training procedure of generative adversarial networks [5] by replacing the implicit generative network with a domain-based scientific simulator, and solve the resulting non-differentiable minimax problem by minimizing variational upper bounds [13, 14] of the adversarial objectives. The procedure results in learning a proposal distribution over simulator parameters, hence producing an arbitrarily tight family of models whose joint collection of generated samples matches the observed data.

II. PROBLEM STATEMENT

We consider a family of parameterized densities $p(\mathbf{x}|\theta)$ defined implicitly through the simulation of a stochastic generative process, where $\mathbf{x} \in \mathbb{R}^d$ is the data and θ are the parameters of interest. The simulation may involve some complicated latent process, such that

$$p(\mathbf{x}|\theta) = \int p(\mathbf{x}|\mathbf{z}, \theta) p(\mathbf{z}|\theta) d\mathbf{z} \tag{1}$$

where $\mathbf{z} \in \mathcal{Z}$ is a latent variable providing an external source of randomness. In particular, $\mathbf{z}$ is not necessarily assumed to be a fixed-size vector (e.g., it can be a sequence of variable length) and its distribution $p(\mathbf{z}|\theta)$ may itself depend on θ in some intricate way.

We assume that we already have an accurate simulation of the stochastic generative process that defines $p(\mathbf{x}|\mathbf{z}, \theta)$, as specified through a highly regularized deterministic function $g(\cdot; \theta) : \mathcal{Z} \to \mathbb{R}^d$ with usually few parameters. That is, we consider

$$\mathbf{x} \sim p(\mathbf{x}|\theta) \equiv \mathbf{z} \sim p(\mathbf{z}|\theta), \mathbf{x} = g(\mathbf{z}; \theta) \tag{2}$$

such that the likelihood $p(\mathbf{x}|\theta)$ can be rewritten as

$$p(\mathbf{x}|\theta) = \frac{\partial}{\partial x_1} \dots \frac{\partial}{\partial x_d} \int_{\{\mathbf{z}: g(\mathbf{z};\theta) \leq \mathbf{x}\}} p(\mathbf{z}|\theta) \mu(d\mathbf{z}), \tag{3}$$

where μ is a probability measure. Importantly, the simulator g is assumed to be a non-invertible function, that can only be used to generate data in forward mode. For this reason, evaluating the integral in Eqn. 3 is intractable. As commonly found in science, we finally assume the lack of access to or existence of derivatives of g with respect to θ, e.g. as when g is specified as a computer program.

Given some observed data $\{\mathbf{x}_i | i = 1, \dots, N\}$ drawn from the (unknown) true distribution $p_r(\mathbf{x})$, our goal is the inference of the parameters of interest θ^* that minimize the divergence between $p_r(\mathbf{x})$ and the modeled data distribution $p(\mathbf{x}|\theta)$ induced by $g(\cdot; \theta)$ over $\mathbf{z}$. That is,

$$\theta^* = \arg\min_{\theta \in \Theta} \rho(p_r(\mathbf{x}), p(\mathbf{x}|\theta)), \tag{4}$$

where ρ is some distance or divergence.

III. BACKGROUND

A. Generative adversarial networks

Generative adversarial networks (GANs) were first proposed by [5] as a way to build an implicit generative model capable of producing samples from random noise $\mathbf{z}$. More specifically, a generative model $g(\cdot;\theta)$ is pit against an adversarial classifier $d(\cdot;\phi) : \mathbb{R}^d \rightarrow [0,1]$ with parameters ϕ and whose antagonistic objective is to recognize real data $\mathbf{x}$ from generated data $\tilde{\mathbf{x}} = g(\mathbf{z};\theta)$. Both models g and d are trained simultaneously, in such a way that g learns to fool its adversary d (which happens when g produces samples comparable to the observed data), while d continuously adapts to changes in g. When d is trained to optimality before each parameter update of the generator, it can be shown that the original adversarial learning procedure amounts to minimizing the Jensen-Shannon divergence $\text{JSD}(p_r(\mathbf{x}) \parallel p(\mathbf{x}|\theta))$ between $p_r(\mathbf{x})$ and $p(\mathbf{x}|\theta)$.

As thoroughly explored in [1], GANs remain remarkably difficult to train because of vanishing gradients as d saturates, or because of unreliable updates when the training procedure is relaxed. As a remedy, Wasserstein GANs [2] reformulate the adversarial setup in order to minimize the Wasserstein-1 distance $W(p_r(\mathbf{x}), p(\mathbf{x}|\theta))$ by replacing the adversarial classifier with a 1-Lipschitz adversarial critic $d(\cdot;\phi) : \mathbb{R}^d \rightarrow \mathbb{R}$. Under the WGAN-GP formulation of [7] for stabilizing the optimization procedure, training d and g results in alternating gradient updates on ϕ and θ in order to respectively minimize

$$\begin{aligned}\mathcal{L}_d =& \mathbb{E}_{\tilde{\mathbf{x}}\sim p(\mathbf{x}|\theta)}[d(\tilde{\mathbf{x}};\phi)] - \mathbb{E}_{\mathbf{x}\sim p_r(\mathbf{x})}[d(\mathbf{x};\phi)] \\ &+ \lambda \mathbb{E}_{\hat{\mathbf{x}}\sim p(\hat{\mathbf{x}})}[(||\nabla_{\hat{\mathbf{x}}} d(\hat{\mathbf{x}};\phi)||_2 - 1)^2] \quad (5)\\ \mathcal{L}_g =& -\mathbb{E}_{\tilde{\mathbf{x}}\sim p(\mathbf{x}|\theta)}[d(\tilde{\mathbf{x}};\phi)] \quad (6)\end{aligned}$$

where $\hat{\mathbf{x}} := \epsilon\mathbf{x} + (1-\epsilon)\tilde{\mathbf{x}}$, for $\epsilon \sim U[0,1]$, $\mathbf{x} \sim p_r(\mathbf{x})$ and $\tilde{\mathbf{x}} \sim p(\mathbf{x}|\theta)$.

B. Variational optimization

Variational optimization [13] and evolution strategies [14] are general optimization techniques that can be used to form a differentiable bound on the optima of a non-differentiable function. Given a function f to minimize, these techniques are based on the simple fact that

$$\min_{\theta\in\Theta} f(\theta) \leq \mathbb{E}_{\theta\sim q(\theta|\psi)}[f(\theta)] = U(\psi), \quad (7)$$

where $q(\theta|\psi)$ is a proposal distribution with parameters ψ over input values θ. That is, the minimum of a set of function values is always less than or equal to any of their average. Provided that the proposal is flexible enough, the parameters ψ can be updated to place its mass arbitrarily tight around the optimum $\theta^* = \min_{\theta\in\Theta} f(\theta)$.

Under mild restrictions outlined in [13], the bound $U(\psi)$ is differentiable with respect to ψ, and using the log-likelihood trick it comes:

$$\begin{aligned}\nabla_\psi U(\psi) &= \nabla_\psi \mathbb{E}_{\theta\sim q(\theta|\psi)}[f(\theta)] \\ &= \nabla_\psi \int f(\theta) q(\theta|\psi) d\theta \\ &= \int f(\theta) \nabla_\psi q(\theta|\psi) d\theta \\ &= \int [f(\theta) \nabla_\psi \log q(\theta|\psi)]\, q(\theta|\psi) d\theta \\ &= \mathbb{E}_{\theta\sim q(\theta|\psi)}[f(\theta)\nabla_\psi \log q(\theta|\psi)] \quad (8)\end{aligned}$$

Effectively, this means that provided that the score function $\nabla_\psi \log q(\theta|\psi)$ of the proposal is known and that one can evaluate $f(\theta)$ for any θ, then one can construct empirical estimates of Eqn. 8, which can in turn be used to minimize $U(\psi)$ with stochastic gradient descent (or a variant thereof, like Adam [10] or the Natural Evolution Strategy algorithm [14], for scaling invariance and robustness to noisy gradients).

IV. ADVERSARIAL VARIATIONAL OPTIMIZATION

The alternating stochastic gradient descent on $\mathcal{L}_d$ and $\mathcal{L}_g$ in GANs (Section III A) inherently assumes that the generator g is a differentiable function. In the setting where we are not interested in learning the implicit model itself but are rather interested in the inference of parameters of a fixed non-differentiable simulator (Section II), gradients $\nabla_\theta g$ either do not exist or cannot be accessed. As a result, gradients $\nabla_\theta \mathcal{L}_g$ cannot be constructed and the optimization procedure cannot be carried out.

In this work, we propose to rely on variational optimization to minimize $\mathcal{L}_d$ and $\mathcal{L}_g$, thereby bypassing the non-differentiability of g. More specifically, we consider a proposal distribution $q(\theta|\psi)$ over the parameters of g and $p(\mathbf{x}|\theta)$ and minimize in alternation the variational upper bounds

$$U_d = \mathbb{E}_{\theta\sim q(\theta|\psi)}[\mathcal{L}_d] \quad (9)$$

$$U_g = \mathbb{E}_{\theta\sim q(\theta|\psi)}[\mathcal{L}_g] \quad (10)$$

respectively over ϕ and ψ. When updating ϕ, unbiased estimates of $\nabla_\phi U_d$ can be obtained by evaluating the exact and known gradient of U_d over mini-batches of true and generated data, as ordinarily done in stochastic gradient descent. When updating ψ, estimates of $\nabla_\psi U_g$ can be derived with forward simulations, as described in the previous section. That is,

$$\nabla_\psi U_g = \mathbb{E}_{\theta\sim q(\theta|\psi), \mathbf{z}\sim p(\mathbf{z}|\theta)}[-d(g(\mathbf{z};\theta);\phi)\nabla_\psi \log q(\theta|\psi)], \quad (11)$$

which we can approximate with mini-batches of generated data

$$\nabla_\psi U_g \approx \frac{1}{M}\sum_{m=1}^{M} -d(g(\mathbf{z}_m;\theta_m);\phi)\nabla_\psi \log q(\theta_m|\psi) \quad (12)$$

Algorithm 1 Adversarial variational optimization.

Inputs: observed data $\{\mathbf{x}_i \sim p_r(\mathbf{x})\}_{i=1}^N$, simulator g.
Outputs: proposal distribution $q(\theta|\psi)$, such that $p_r(\mathbf{x}) \approx p(\mathbf{x}|\psi)$.
Hyper-parameters: The number n_{critic} of training iterations of d; the size M of a mini-batch; the gradient penalty coefficient λ; the entropy penalty coefficient γ.

$q(\theta|\psi) \leftarrow$ prior on θ (with differentiable and known density)
while ψ has not converged **do**
  for $i = 1$ to n_{critic} **do** ▷ Update d
    Sample a mini-batch $\{\mathbf{x}_m \sim p_r(\mathbf{x}), \theta_m \sim q(\theta|\psi), \mathbf{z}_m \sim p(\mathbf{z}|\theta_m), \epsilon_m \sim U[0,1]\}_{m=1}^M$.
    for $m = 1$ to M **do**
      $\tilde{\mathbf{x}}_m \leftarrow g(\mathbf{z}_m; \theta_m)$
      $\hat{\mathbf{x}}_m \leftarrow \epsilon_m \mathbf{x}_m + (1 - \epsilon_m)\tilde{\mathbf{x}}_m$
      $U_d^{(m)} \leftarrow d(\tilde{\mathbf{x}}_m; \phi) - d(\mathbf{x}_m; \phi) + \lambda(||\nabla_{\hat{\mathbf{x}}_m} d(\hat{\mathbf{x}}_m; \phi)||_2 - 1)^2$
    end for
    $\phi \leftarrow \text{Adam}(\nabla_\phi \frac{1}{M} \sum_{m=1}^M U_d^{(m)})$
  end for
  Sample a mini-batch $\{\theta_m \sim q(\theta|\psi), \mathbf{z}_m \sim p(\mathbf{z}|\theta_m)\}_{m=1}^M$. ▷ Update $q(\theta|\psi)$
  $\nabla_\psi U_g \leftarrow \frac{1}{M} \sum_{m=1}^M -d(g(\mathbf{z}_m; \theta_m)) \nabla_\psi \log q_\psi(\theta_m)$
  $\nabla_\psi H(q_\psi) \leftarrow \frac{1}{M} \sum_{m=1}^M \nabla_\psi q_\psi(\theta_m) \log q_\psi(\theta_m)$
  $\psi \leftarrow \text{Adam}(\nabla_\psi U_g + \gamma \nabla_\psi H(q_\psi))$
end while

for $\theta_m \sim q(\theta|\psi)$ and $\mathbf{z}_m \sim p(\mathbf{z}|\theta_m)$. For completeness, Algorithm 1 outlines the proposed adversarial variational optimization procedure, as built on top of WGAN-GP. Obviously, the variational relaxation could similarly be coupled with other variants of GANs and/or of evolution strategies.

Practically, the variational objectives 9-10 have the effect of replacing the modeled data distribution of Eqn. 2 with a distribution parameterized in terms of ψ:

$$\mathbf{x} \sim p(\mathbf{x}|\psi) \equiv \theta \sim q(\theta|\psi), \mathbf{z} \sim p(\mathbf{z}|\theta), \mathbf{x} = g(\mathbf{z}; \theta). \quad (13)$$

Intuitively, this corresponds to a family of simulators, each configured with randomly sampled parameters $\theta \sim q(\theta|\psi)$, whose joint collection of generated samples is optimized with adversarial training to approach the real data distribution $p_r(\mathbf{x})$. More formally, the learned model therefore corresponds to the marginal distribution of the generated data. That is,

$$p(\mathbf{x}|\psi) = \int q(\theta|\psi) p(\mathbf{x}|\theta) d\theta. \quad (14)$$

In consequence, the proposed inference algorithm does not necessarily guarantee that the proposal distribution $q(\theta|\psi)i$ will place its mass arbitrarily tight around the parameters of interest, which might be an issue when one is rather interested in point estimates θ^*. For this purpose, we augment Eqn. 10 with a regularization term corresponding to the differential entropy H of the proposal distribution. That is,

$$U_g = \mathbb{E}_{\theta \sim q(\theta|\psi)}[\mathcal{L}_g] + \gamma H(q(\theta|\psi)) \quad (15)$$

where $\gamma \in \mathbb{R}^+$ is a hyper-parameter controlling the trade-off between the generator objective and the tightness of the proposal distribution. For small values of γ, proposal distributions with large entropy are not penalized, which may result in learning a smeared variation of the original simulator. On the other hand, for large values of γ, the procedure is constrained to fit a proposal distribution with low entropy, which has the effect of concentrating its density tightly around one or a few θ values. A too large penalty may however eventually make the optimization unstable, as the variance of $\nabla_\psi \log q(\theta_m|\psi)$ typically increases as the entropy of the proposal decreases.

V. EXPERIMENTS

A. Univariate discrete data

As a first illustrative experiment, we evaluate inference for a discrete Poisson distribution with unknown scale parameter λ. We artificially consider the distribution as a parameterized simulator, from which we can only generate data.

The observed data is sampled from a Poisson with scale $\lambda^* = 7$. Algorithm 1 is run for 300 epochs with mini-batches of size $M = 64$ and the following configuration. For the critic d, we use a 3-layer MLP with 10 hidden nodes per layer and ReLU activations. At each epoch, Adam is run for $n_{\text{critic}} = 100$ iterations with a step size $\alpha = 0.01$, decay rates $\beta_1 = \beta_2 = 0.5$ and its inner first and second moment vectors reset at each outer epoch in order to avoid building momentum in staled directions. For estimating λ^*, we parameterize θ as $\log(\lambda)$ and use a univariate Gaussian proposal distribution $q(\theta|\psi)$ initialized with a mean at $\log(5)$ and unit variance. At each epoch, parameters ψ are updated by taking one Adam

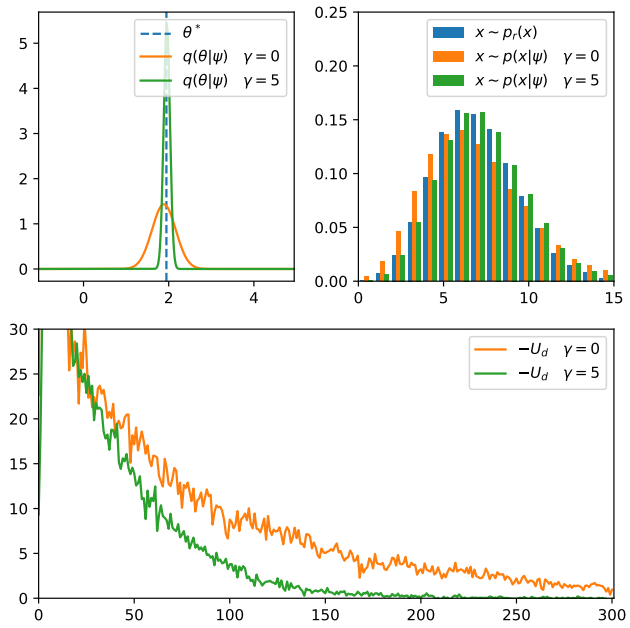

FIG. 1. Discrete Poisson model with unknown scale. (*Top left*) Proposal distributions $p(\theta|\psi)$ after adversarial variational optimization. For both $\gamma = 0$ and $\gamma = 5$, the distributions correctly concentrate their density around the true value $\log(\lambda^*)$. Penalizing the entropy of the proposal distribution ($\gamma = 5$) results in a tighter density and therefore to better inference. (*Top right*) Model distributions $p(\mathbf{x}|\psi)$ after training. This plot shows that the resulting parameterizations of the simulator closely reproduce the true distribution. (*Bottom*) Empirical estimates of the variational upper bound U_d as optimization progresses.

step, with $\alpha = 0.01$ and $\beta_1 = \beta_2 = 0.1$. The gradient penalty coefficient is set to $\lambda = 0.025$, and the entropy penalty is evaluated at both $\gamma = 0$ and $\gamma = 5$.

The top left plot in Figure 1 illustrates the resulting proposal distributions $p(\theta|\psi)$ after adversarial variational optimization. For both $\gamma = 0$ and $\gamma = 5$, the proposal distributions correctly concentrate their density around the true parameter value $\log(\lambda^*) = 1.94$. Under the effect of the positive entropy penalty $H(q(\theta|\psi))$, the proposal distribution for $\gamma = 5$ concentrates its mass more tightly, yielding in this case a more precise inference. The top right plot compares the model distributions to the true distribution. As theoretically expected from adversarial training, we see that the resulting distributions closely match the true distribution, with in this case slightly better results for the penalized model. The bottom plot of Figure 1 shows empirical estimates of $-U_d$ with respect to the epoch number. For both $\gamma = 0$ and $\gamma = 5$, the curves quickly fall towards 0, which indicates that $\mathbb{E}_{\tilde{\mathbf{x}} \sim p(\mathbf{x}|\theta)}[d(\tilde{\mathbf{x}}; \phi)] \approx \mathbb{E}_{\mathbf{x} \sim p_r(\mathbf{x})}[d(\mathbf{x}; \phi)]$ and that the critic cannot distinguish between true and model data. Despite the discreteness and the non-differentiability of the underlying generator, this confirms that inference with adversarial variational optimization works.

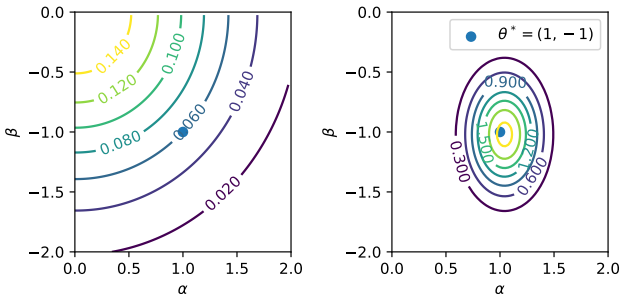

FIG. 2. Multidimensional continuous data. (*Left*) Likelihood $q(\theta|\psi)$ at the beginning of the procedure, for a proposal distribution initialized with zero mean and unit variance. (*Right*) Likelihood $q(\theta|\psi)$ after adverarial variational optimization. The proposal density correctly concentrates around $\theta^* = (1, -1)$.

B. Multidimensional continuous data

As a second example, we evaluate parameter inference for a generator producing 5-dimensional continuous data, as originally specified in Section 4.2. of [3]. More specifically, we consider the following generative process:

- $\mathbf{z} = (z_0, z_1, z_2, z_3, z_4)$, such that $z_0 \sim \mathcal{N}(\mu = \alpha, \sigma = 1)$, $z_1 \sim \mathcal{N}(\mu = \beta, \sigma = 3)$, $z_2 \sim \text{Mixture}(\frac{1}{2}\mathcal{N}(\mu = -2, \sigma = 1), \frac{1}{2}\mathcal{N}(\mu = 2, \sigma = 0.5))$, $z_3 \sim \text{Exponential}(\lambda = 3)$, and $z_4 \sim \text{Exponential}(\lambda = 0.5)$;
- $\mathbf{x} = R\mathbf{z}$, where R is a fixed semi-positive definite 5×5 matrix defining a fixed projection of $\mathbf{z}$ into the observed space.

We consider observed data generated at the nominal values $\theta^* = (\alpha^* = 1, \beta^* = -1)$. The simulator parameters are modeled with a factored Gaussian proposal distribution $q(\theta|\psi) = q(\alpha|\psi)q(\beta|\psi)$, where each component is initialized with zero mean and unit variance. The entropy penalty is set to $\gamma = 5$, while all other hyper-parameters are otherwise the same as in the previous example.

Starting with a proposal distribution $q(\theta|\psi)$ largely spread over the parameter space, as illustrated in the left plot of Figure 2, inference quickly converges towards a proposal distribution whose density concentrates around the nominal values θ^*, as shown in the right plot of Figure 2.

C. Weinberg

VI. RELATED WORKS

As reviewed in [12], likelihood-free inference is intimately tied to a class of algorithms that can be framed as density estimation-by-comparison. In most cases, these

inference algorithms are formulated as an iterative two-step process where the model distribution is first compared to the true data distribution and then updated to make it more comparable to the latter.

Closest to our work are procedures that rely on a classifier to estimate the discrepancy between the true and the model distributions. For example, [9] uses non linear logistic regression for fitting unnormalized differentiable statistical models, while [5] exploits an adversarial neural network for learning a differentiable implicit generative model. In the likelihood-free setup, [3, 4] estimate likelihood ratios through supervised classification, which can in turn be used for parameter inference in combination with a gradient-free optimization algorithm. Similarly, [8] makes use of classification accuracy as a summary statistics for approximate Bayesian computation.

In this context, the proposed method can be considered as a direct adaptation of generative adversarial networks [5] to non-differentiable simulators. It also constitutes an approximate gradient descent alternative to bayesian optimization based on density ratios [3] or to classifier ABC [8].

VII. SUMMARY

[GL: avo = extension of gans to non-differentiable generator] [GL: nice for discrete data] [GL: next version = try on an actual particle physics simulator.]

[GL: Can we exploit the fact that $\nabla_{\mathbf{x}} d(\mathbf{x})$ is known exactly for building better estimates $\tilde{\nabla}_{\psi} U_g$?]

ACKNOWLEDGMENTS

GL and KL are both supported through NSF ACI-1450310, additionally KC is supported through PHY-1505463 and PHY-1205376.

[1] Arjovsky, M., and Bottou, L. Towards Principled Methods for Training Generative Adversarial Networks. *ArXiv e-prints* (Jan. 2017).

[2] Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein GAN. *ArXiv e-prints* (Jan. 2017).

[3] Cranmer, K., Pavez, J., and Louppe, G. Approximating likelihood ratios with calibrated discriminative classifiers. *arXiv preprint arXiv:1506.02169* (2015).

[4] Dutta, R., Corander, J., Kaski, S., and Gutmann, M. U. Likelihood-free inference by ratio estimation. *ArXiv e-prints* (Nov. 2016).

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

[6] Graham, M. M., and Storkey, A. J. Asymptotically exact inference in differentiable generative models. *ArXiv e-prints* (May 2016).

[7] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. Improved Training of Wasserstein GANs. *ArXiv e-prints* (Mar. 2017).

[8] Gutmann, M. U., Dutta, R., Kaski, S., and Corander, J. Likelihood-free inference via classification. *Statistics and Computing* (2017), 1–15.

[9] Gutmann, M. U., and Hyvärinen, A. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research 13*, Feb (2012), 307–361.

[10] Kingma, D. P., and Ba, J. Adam: A Method for Stochastic Optimization. *ArXiv e-prints* (Dec. 2014).

[11] Le, T. A., Gunes Baydin, A., and Wood, F. Inference Compilation and Universal Probabilistic Programming. *ArXiv e-prints* (Oct. 2016).

[12] Mohamed, S., and Lakshminarayanan, B. Learning in Implicit Generative Models. *ArXiv e-prints* (Oct. 2016).

[13] Staines, J., and Barber, D. Variational Optimization. *ArXiv e-prints* (Dec. 2012).

[14] Wierstra, D., Schaul, T., Glasmachers, T., Sun, Y., and Schmidhuber, J. Natural Evolution Strategies. *ArXiv e-prints* (June 2011).