# 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 likehood function for a given observation. In this note, we develop a likelihood-free inference algorithm for fitting a forward non-differentiable generative model to 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 with variational optimization. Effectively, the adversarial variational optimization procedure results in learning an arbitrarily tight proposal distribution over simulator parameters, such that the respective hierarchical model matches the observed data. [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 [6] are specified as imperative implementations of forward stochastic processes that generate 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 this note, we develop a likelihood-free inference algorithm for fitting a forward non-differentiable generative model to observed data. We propose to adapt the adversarial training procedure of generative adversarial networks [4] by replacing the implicit generative network with a domain-based scientific simulator, and solve the resulting non-differentiable minimax problem with variational optimization [7]. The adversarial variational optimization procedure results in learning a proposal distribution over simulator parameters, hence producing an abitrarily tight family of models whose joint collection of generated samples matches the observed data.

## II. PROBLEM STATEMENT

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

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

where $\mathbf{z} \in \mathbb{R}^m$ is a latent variable providing an external source of randomness.

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

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

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

$$p_\theta(\mathbf{x}) = \frac{\partial}{\partial x_1} \cdots \frac{\partial}{\partial x_d} \int_{\{\mathbf{z}:g(\mathbf{z};\theta)\leq \mathbf{x}\}} p(\mathbf{z})d\mathbf{z}. \tag{3}$$

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 $\theta$, e.g. as when $g$ is specified as a computer program.

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

$$\theta^* = \arg\min_\theta \rho(p_r, p_\theta), \tag{4}$$

where $\rho$ is some distance or divergence.

## III. BACKGROUND

### A. Generative adversarial networks

Generative adversarial networks (GANs) were first proposed by [4] 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 $d(\cdot;\phi) : \mathbb{R}^d \to [0,1]$ with parameters $\phi$ 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 maximally confuse 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 $\text{JSD}(p_r \parallel p_\theta)$ between $p_r$ and $p_\theta$.

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 $W(p_r, p_\theta)$ by replacing the adversarial classifier with a 1-Lipschitz adversarial critic $d(\cdot;\phi) : \mathbb{R}^d \to \mathbb{R}$. Under the WGAN-GP formulation of [5] for stabilizing the optimization procedure, training $d$ and $g$ results in alternating gradient updates on $\phi$ and $\theta$ in order to respectively minimize

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

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

## B. Variational optimization

Following [7], variational optimization (VO) (also known as the search gradient algorithm [8] related to evolution strategies) is a general optimization technique that can be used to form a differentiable bound on the optima of a non-differentiable function. Given a function $f$ to minimize, VO is based on the simple fact that

$$\min_{\mathbf{c}\in\mathcal{C}} f(\mathbf{c}) \leq \mathbb{E}_{\mathbf{c}\sim q_\psi(\mathbf{c})}[f(\mathbf{c})] = U(\psi), \qquad (7)$$

where $q_\psi$ is a proposal distribution with parameters $\psi$ over input values $\mathbf{c}$. 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 $\psi$ can be updated to place its mass arbitrarily tight around the optimum $\mathbf{c}^* = \min_{\mathbf{c}\in\mathcal{C}} f(\mathbf{c})$.

Under mild restrictions outlined in [7], the bound $U(\psi)$ is differentiable, and using the log-likelihood trick it comes:

$$\begin{aligned}\nabla_\psi U(\psi) &= \nabla_\psi \mathbb{E}_{\mathbf{c}\sim q_\psi(\mathbf{c})}[f(\mathbf{c})] \\ &= \nabla_\psi \int f(\mathbf{c}) q_\psi(\mathbf{c}) d\mathbf{c} \\ &= \int f(\mathbf{c}) \nabla_\psi q_\psi(\mathbf{c}) d\mathbf{c} \\ &= \int [f(\mathbf{c}) \nabla_\psi \log q_\psi(\mathbf{c})]\, q_\psi(\mathbf{c}) d\mathbf{c} \\ &= \mathbb{E}_{\mathbf{c}\sim q_\psi(\mathbf{c})}[f(\mathbf{c}) \nabla_\psi \log q_\psi(\mathbf{c})] \qquad (8)\end{aligned}$$

Effectively, this means that provided that the score function $\nabla_\psi \log q_\psi(\mathbf{c})$ of the proposal is known and that one can evaluate $f(\mathbf{c})$ for any $\mathbf{c}$, then one can construct empirical estimates of Eqn. 8, which can in turn be used to perform stochastic gradient descent (or a variant thereof) in order to minimize $U(\psi)$.

## 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 perform variational optimization on $\mathcal{L}_d$ and $\mathcal{L}_g$, thereby bypassing the non-differentiability of $g$. More specifically, we consider a proposal distribution $q_\psi(\theta)$ over the parameters of $g$ and minimize in alternance the variational upper bounds

$$\begin{aligned}U_d &= \mathbb{E}_{\theta\sim q_\psi}[\mathcal{L}_d] \qquad (9)\\ U_g &= \mathbb{E}_{\theta\sim q_\psi}[\mathcal{L}_g] \qquad (10)\end{aligned}$$

respectively over $\phi$ and $\psi$. When updating $d$, unbiased gradient 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 $g$, gradient estimates of $\nabla_\psi U_g$ can be derived with forward simulations, as described in Eqn. 8. That is,

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

which we can approximate with mini-batches of generated data

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

for $\theta_m \sim q_\psi$ and $\mathbf{z}_m \sim p_z$. [GL: Can we exploit the fact that $\nabla_{\mathbf{x}} d(\mathbf{x})$ is known exactly for building better estimates $\tilde{\nabla}_\psi U_g$? In practice, using

$$\tilde{\nabla}_\psi U_g = \frac{1}{M}\sum_{m=1}^{M} g(\mathbf{z}_m;\theta_m)\nabla_\psi \log q_\psi(\theta_m)\nabla_{\tilde{\mathbf{x}}}(-d(\tilde{\mathbf{x}}_m;\phi)) \quad (13)$$

works as good it seems, but I am not sure to precisely understand why, nor whether this is correct... It feels like this is approximating the chain rule $\nabla_\theta g(\theta)\nabla_{\tilde{\mathbf{x}}}(-d(\tilde{\mathbf{x}}))$. ]

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 $\psi$:

$$\mathbf{x} \sim p_\psi \equiv \mathbf{z} \sim p_z, \theta \sim q_\psi, \mathbf{x} = g(\mathbf{z};\theta). \qquad (14)$$

**Algorithm 1** Adversarial variational optimization.

*Inputs:* observed data $\{\mathbf{x}_i\}_{i=1}^N$, simulator $g$;
*Outputs:* proposal distribution $q_\psi(\theta)$;
*Hyper-parameters:* The number $n_{\text{critic}}$ of training iterations of $d$; the gradient penalty coefficient $\lambda$; the entropy penalty coefficient $\gamma$.

```
1:  while $\psi$ has not converged do
2:      for $i = 1$ to $n_{\text{critic}}$ do                                         ▷ Update $d$
3:          Sample a mini-batch $\{\mathbf{x}_m\}_{m=1}^M \sim p_r$, $\{\mathbf{z}_m\}_{m=1}^M \sim p_z$, $\{\theta_m\}_{m=1}^M \sim q_\psi$, $\{\epsilon_m\}_{m=1}^M \sim U[0,1]$.
4:          for $m = 1$ to $M$ do
5:              $\tilde{\mathbf{x}}_m \leftarrow g(\mathbf{z}_m; \theta_m)$
6:              $\hat{\mathbf{x}}_m \leftarrow \epsilon_m \mathbf{x}_m + (1-\epsilon_m)\tilde{\mathbf{x}}_m$
7:              $L^{(m)} \leftarrow d(\tilde{\mathbf{x}}_m; \phi) - d(\mathbf{x}_m; \phi) + \lambda(||\nabla_{\hat{\mathbf{x}}_m} d(\hat{\mathbf{x}}_m; \phi)||_2 - 1)^2$
8:          end for
9:          $\phi \leftarrow \text{RMSProp}(\nabla_\phi \frac{1}{M}\sum_{m=1}^M L^{(m)})$
10:     end for
11:     Sample a mini-batch $\{\mathbf{z}_m\}_{m=1}^M \sim p_z$, $\{\theta_m\}_{m=1}^M \sim q_\psi$.                ▷ Update $q_\psi$
12:     $\psi \leftarrow \text{RMSProp}(\frac{1}{M}\sum_{m=1}^M -d(g(\mathbf{z}_m;\theta_m))\nabla_\psi \log q_\psi(\theta_m) + \nabla_\psi \gamma q_\psi(\theta_m)\log q_\psi(\theta_m))$
13: end while
```

Intuitively, this corresponds to a family of simulators, each configured with randomly sampled parameters $\theta \sim q_\psi$, whose joint collection of generated samples is optimized with adversarial training to approach the real data distribution $p_r$. However, this formulation does not necessarily guarantee that the proposal distribution $q_\psi$ will place its mass arbitrarily tight around the parameters of interest $\theta^*$. For this purpose, we augment Eqn. 10 with a regularization term corresponding to the differential entropy $H$ of the proposal distribution. That is,

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

where $\gamma \in \mathbb{R}^+$ is a hyperparameter controlling the trade-off between the generator objective and the tightness of the proposal distribution. For large values of $\gamma$, 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 $\theta$ values. On the other hand, for small values of $\gamma$, proposal distributions with larger entropy are not penalized, which may result in learning a smeared variation of the original simulator.

For completeness, Algorithm 1 finally outlines the adversarial variational optimization procedure when coupled with WGAN-GP.

## V. EXPERIMENTS

### A. Toy problem

### B. Physics example

## VI. RELATED WORKS

[GL: Implicit generative models.] [GL: gan + classifier ABC.]

[GL: likelihood free inference, density ratio by classification] [GL: carl [3].]

## VII. SUMMARY

## 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] 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.

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

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

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

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