Probability Functional Descent: A Unifying Perspective on GANs, Variational Inference, and Reinforcement Learning

Casey Chu [1] **Jose Blanchet** [2] **Peter Glynn** [2]

Abstract

The goal of this paper is to provide a unifying view of a wide range of problems of interest in machine learning by framing them as the minimization of functionals defined on the space of probability measures. In particular, we show that generative adversarial networks, variational inference, and actor-critic methods in reinforcement learning can all be seen through the lens of our framework. We then discuss a generic optimization algorithm for our formulation, called *probability functional descent* (PFD), and show how this algorithm recovers existing methods developed independently in the settings mentioned earlier.

1. Introduction

Deep learning now plays an important role in many domains, for example, in generative modeling, deep reinforcement learning, and variational inference. In the process, dozens of new algorithms have been proposed for solving these problems with deep neural networks, specific of course to domain at hand.

In this paper, we introduce a conceptual framework which can be used to understand in a unified way a broad class of machine learning problems. Central to this framework is an abstract optimization problem in the space of probability measures, a formulation that stems from the observation that in many fields, the object of interest is a probability distribution; moreover, the learning process is guided by a *probability functional* to be minimized, a loss function that conceptually maps a probability distribution to a real number. Table 1 lists these correspondences in the case of generative adversarial networks, variational inference, and reinforcement learning.

Because the optimization now takes place in the infinite-dimensional space of probability measures, standard finite-dimensional algorithms like gradient descent are initially unavailable; even the proper notion for the derivative of these functionals is unclear. We call upon on a body of literature known as von Mises calculus (von Mises, 1947; Fernholz, 2012), originally developed in the field of asymptotic statistics, to make these functional derivatives precise. Remarkably, we find that once the connection is made, the resulting generalized descent algorithm, which we call *probability functional descent*, is intimately compatible with standard deep learning techniques such as stochastic gradient descent (Bottou, 2010), the reparameterization trick (Kingma & Welling, 2013), and adversarial training (Goodfellow et al., 2014).

When we apply probability functional descent to the aforementioned domains, we find that we recover a wide range of existing algorithms, and the essential distinction between them is simply the way that the functional derivative, the *von Mises influence function* in this context, is approximated. Table 2 lists these algorithms and their corresponding approximation methods. Probability functional descent therefore acts as a unifying framework for the analysis of existing algorithms as well as the systematic development of new ones.

1.1. Related work

The problem of optimizing functionals of probability measures is not new. For example, Gaivoronski (1986) and Molchanov & Zuyev (2001) study these types of problems and even propose Frank-Wolfe and steepest descent algorithms to solve these problems. However, their algorithms are not immediately practical for the high-dimensional machine learning settings described here, and it is not clear how to integrate their methods with modern deep learning techniques.

Several others in the machine learning community also adopt the perspective of descent in the space of probability distributions. In order to introduce functional gradients, these approaches endow the space of probability distribu-

[1]Institute for Computational & Mathematical Engineering, Stanford University, Stanford, California, USA [2]Management Science & Engineering, Stanford University, Stanford, California, USA. Correspondence to: Casey Chu <caseychu@stanford.edu>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

Domain	Distribution of interest	Functional	Functional derivative
Generative adversarial networks	Generator μ	$D(\mu\|\|\nu)$	Discriminator $D^*(x)$
Variational inference	Approximate posterior $q(z)$	$D_{\mathrm{KL}}(q(z)\|\|p(z\|x))$	Negative ELBO $\log \frac{q(z)}{p(x,z)}$
Reinforcement learning	Policy $\pi(a\|s)$	Expected reward	Advantage $Q^\pi(s,a) - V^\pi(s)$

Table 1. Framing a problem as the optimization of a probability functional unifies several domains.

Algorithm	Type of derivative estimator
Generative adversarial networks	
Minimax GAN (Goodfellow et al., 2014)	Convex duality
Non-saturating GAN (Goodfellow et al., 2014)	Binary classification
Wasserstein GAN (Arjovsky et al., 2017)	Convex duality
Variational inference	
Black-box variational inference (Ranganath et al., 2014)	Exact
Adversarial variational Bayes (Mescheder et al., 2017)	Binary classification
Adversarial posterior distillation (Wang et al., 2018)	Convex duality
Reinforcement learning	
Policy iteration (Howard, 1960)	Exact
Policy gradient (Williams, 1992)	Monte Carlo
Actor-critic (Konda & Tsitsiklis, 2000; Sutton et al., 2000)	Least squares
Dual actor-critic (Chen & Wang, 2016; Dai et al., 2017b)	Convex duality

Table 2. Different existing algorithms correspond to different ways of estimating the functional derivative.

tions with either Hilbert structure (Dai et al., 2014; 2016; Liu & Wang, 2016; Dai, 2018) or Wasserstein structure (Richemond & Maginnis, 2017; Frogner & Poggio, 2018; Zhang et al., 2018; Lin et al., 2018) and rely on gradient descent or Wasserstein gradient flow respectively to decrease the objective value. Such approaches typically require kernel-based or particle-based methods to implement in practice. By contrast, our approach foregoes gradients and instead directly considers descent on linear approximations by leveraging the Gâteaux derivative. As we shall illustrate, this approach is more compatible with standard deep learning techniques and indeed leads exactly to many existing deep learning-based algorithms. Carmona & Delarue (2018) provide a technical comparison between these differing approaches for defining derivatives in chapter 5.

Finally, one part of our work recasts convex optimization problems as saddle-point problems by means of convex duality as a technique for estimating functional derivatives. This correspondence between convex optimization problems and saddle point problems is an old and general concept (Rockafellar, 1968), and it underlies classical dual optimization techniques (Lucchetti, 2006; Luenberger & Ye, 2015). Nevertheless, the use of these min-max representations remains an active topic of research in machine learning. Most notably, the literature concerning generative adversarial networks has recognized that certain min-max problems are equivalent to certain convex problems (Goodfellow et al., 2014; Nowozin et al., 2016; Farnia & Tse, 2018). Outside of GANs, Dai et al. (2017a; 2018) have begun using these min-max representations to inspire learning algorithms. These min-max representations are an important tool for us that allows for practical implementation of our theory.

2. Descent on a Probability Functional

We let $\mathcal{P}(X)$ be the space of Borel probability measures on a topological space X. Our abstract formulation takes the form of a minimization problem over probability distributions:

$$\min_{\mu \in \mathcal{P}(X)} J(\mu),$$

where $J : \mathcal{P}(X) \to \mathbb{R}$ is called a probability functional. In order to avoid technical digressions, we assume that X is a metric space that is compact, complete, and separable (i.e. a compact Polish space). We endow $\mathcal{P}(X)$ with the topology of weak convergence, also known as the weak* topology.

We now draw upon elements of von Mises calculus (von Mises, 1947) to make precise the notion of derivatives of functionals such as J. See Fernholz (2012) for an in-depth discussion, or Santambrogio (2015) for another perspective.

Definition 1 (Gâteaux differential). *Let $J : \mathcal{P}(X) \to \mathbb{R}$ be a function. The Gâteaux differential dJ_μ at $\mu \in \mathcal{P}(X)$*

in the direction χ is defined by

$$dJ_\mu(\chi) = \lim_{\epsilon \to 0^+} \frac{J(\mu + \epsilon\chi) - J(\mu)}{\epsilon}, \tag{1}$$

where $\chi = \nu - \mu$ for some $\nu \in \mathcal{P}(X)$.

Intuitively, the Gâteaux differential is a generalization of the directional derivative, so that $dJ_\mu(\chi)$ describes the change in the value of $J(\mu)$ when the probability measure μ is infinitesimally perturbed in the direction of χ, towards another measure ν. Though powerful, the Gâteaux differential is a function of differences of probability measures, which can make it unwieldy to work with. In many cases, however, the Gâteaux differential $dJ_\mu(\chi)$ can be concisely represented as an integral of an *influence function* $\Psi_\mu : X \to \mathbb{R}$, where the integral is taken with respect to the measure χ.

Definition 2 (Influence function). *We say that $\Psi_\mu : X \to \mathbb{R}$ is an influence function for J at $\mu \in \mathcal{P}(X)$ if the Gâteaux differential $dJ_\mu(\chi)$ has the integral representation*

$$dJ_\mu(\chi) = \int_X \Psi_\mu(x)\, \chi(dx) \tag{2}$$

for all $\chi = \nu - \mu$, where $\nu \in \mathcal{P}(X)$.

The influence function provides a convenient representation for the Gâteaux differential. Because $\chi = \nu - \mu$ is a difference of probability distributions, we can also write

$$dJ_\mu(\chi) = \mathbb{E}_{x\sim\nu}[\Psi_\mu(x)] - \mathbb{E}_{x\sim\mu}[\Psi_\mu(x)]$$

by linearity. We note that if Ψ_μ is an influence function, then so is $\Psi_\mu + c$ for a constant c.

The Gâteaux derivative and the influence function provide the proper notion of a functional derivative, which allows us to generalize first-order descent algorithms to apply to probability functionals such as J. In particular, they permit a linear approximation to $J(\mu)$ around μ_0, which we denote $\tilde{J}(\mu)$:

$$\begin{aligned}\tilde{J}(\mu) &= J(\mu_0) + dJ_{\mu_0}(\mu - \mu_0) \\ &= J(\mu_0) + \mathbb{E}_{x\sim\mu}[\Psi_{\mu_0}(x)] - \mathbb{E}_{x\sim\mu_0}[\Psi_{\mu_0}(x)] \\ &= \text{constant} + \mathbb{E}_{x\sim\mu}[\Psi_{\mu_0}(x)].\end{aligned}$$

This expression, also known as a von Mises representation, yields additional intuition about the influence function. Concretely, note that a small pertubation to μ decreases $J(\mu)$ if it decreases $\mathbb{E}_{x\sim\mu}[\Psi_{\mu_0}(x)]$. Therefore, Ψ_{μ_0} acts as a potential function defined on X that dictates where samples $x \sim \mu$ should descend if the goal is to decrease $J(\mu)$. Of course, Ψ_{μ_0} only carries this interpretation around the current value of μ_0.

Based on this intuition, we now present **probability functional descent**, a straightforward analogue of finite-dimensional first-order descent algorithms to probability functionals. First, a linear approximation to the functional J is computed at μ_0 in the form of the influence function Ψ_{μ_0}, and then a local step is taken from μ_0 so as to decrease the value of the linear approximation. Concretely:

Algorithm 1 Probability functional descent on $J(\mu)$

Initialize μ to a distribution in $\mathcal{P}(X)$
while μ has not converged **do**
  Set $\hat{\Psi} \approx \Psi_\mu$ (differentiation step)
  Update μ to decrease $\mathbb{E}_{x\sim\mu}[\hat{\Psi}(x)]$ (descent step)
end while

We shall see that probability functional descent serves as a blueprint for many existing algorithms: in generative adversarial networks, the differentiation and descent steps correspond to the discriminator and generator updates respectively; in reinforcement learning, they correspond to policy evaluation and policy improvement.

In its abstract form, probability functional descent requires two design choices in order to convert it into a practical algorithm. In section 3, we discuss different ways to choose the update in the descent step; Theorem 1 provides one generic way. In section 4, we discuss different ways to approximate the influence function in the differentiation step; Theorem 2 provides one generic way and an unexpected connection to adversarial training.

3. Applying the Descent Step

One straightforward way to apply the descent step of PFD is to adopt a parametrization $\theta \mapsto \mu_\theta$ and descend the stochastic gradient of $\theta \mapsto \mathbb{E}_{x\sim\mu_\theta}[\hat{\Psi}(x)]$.[1] This gradient step is justified by the following analogue of the chain rule:

Theorem 1 (Chain rule). *Let $J : \mathcal{P}(X) \to \mathbb{R}$ be continuously differentiable, in the sense that the influence function Ψ_μ exists and $(\mu, \nu) \mapsto \mathbb{E}_{x\sim\nu}[\Psi_\mu(x)]$ is continuous. Let the parameterization $\theta \mapsto \mu_\theta$ be differentiable, in the sense that $\frac{1}{||h||}(\mu_{\theta+h} - \mu_\theta)$ converges to a weak limit as $h \to 0$. Then*

$$\nabla_\theta J(\mu_\theta) = \nabla_\theta \mathbb{E}_{x\sim\mu_\theta}[\hat{\Psi}(x)],$$

where $\hat{\Psi} = \Psi_{\mu_\theta}$ is treated as a function $X \to \mathbb{R}$ that is not dependent on θ.

Theorem 1 converts the computation of $\nabla_\theta J(\mu_\theta)$, where J may be a complicated nonlinear functional, into the computation of a gradient of an expectation, which is easily handled using standard methods (see e.g. Schulman et al. (2015)). For example, the reparameterization trick, also

[1] Note that this gradient step is simply one possible choice of update rule for the descent step of PFD; see subsection 7.1 (policy iteration) for an instance of PFD where this gradient-based update rule is not adopted.

known as the pathwise derivative estimator (Kingma & Welling, 2013; Rezende et al., 2014), uses the identity

$$\nabla_\theta \mathbb{E}_{x\sim\mu_\theta}[\hat{\Psi}(x)] = \nabla_\theta \mathbb{E}_{z\sim\mathcal{N}(0,I)}[\hat{\Psi}(h_\theta(z))],$$

where μ_θ samples $x = h_\theta(z)$ using $z \sim \mathcal{N}(0, I)$. Alternatively, the log derivative trick, also known as the score function gradient estimator, likelihood ratio gradient estimator, or REINFORCE (Glynn, 1990; Williams, 1992; Kleijnen & Rubinstein, 1996), uses the identity

$$\nabla_\theta \mathbb{E}_{x\sim\mu_\theta}[\hat{\Psi}(x)] = \mathbb{E}_{x\sim\mu_\theta}[\hat{\Psi}(x)\nabla_\theta \log \mu_\theta(x)],$$

where $\mu_\theta(x)$ is the probability density function of μ_θ. This gradient-based update rule for the descent step is therefore a natural, practical choice in the context of deep learning.

4. Approximating the Influence Function

The approximation of the influence function in the differentiation step can in principle be accomplished in many different ways. Indeed, we shall see that the distinguishing factor between many existing algorithms is exactly which influence function estimator used, as shown in Table 2. In some cases, it is possible that the influence function can be evaluated exactly, bypassing the need for approximation. Otherwise, the influence function, being a function $X \to \mathbb{R}$, may be modeled as a neural network; the precise way in which this neural network needs to be trained will depend on the exact analytical form of the influence function.

Remarkably, a generic approximation technique is available if the functional J is convex. In this case, the influence function Ψ_μ possesses a variational characterization in terms of the convex conjugate $J^\star$ of J. To apply this formalism, we now view $\mathcal{P}(X)$ as a convex subset of the vector space of finite signed Borel measures $\mathcal{M}(X)$, equipped with the topology of weak convergence. Crucial to the analysis will be its dual space, $\mathcal{C}(X)$, the space of continuous functions $X \to \mathbb{R}$. Finally, $\overline{\mathbb{R}}$ denotes the extended real line $\mathbb{R} \cup \{-\infty, \infty\}$. The convex conjugate is then defined as follows:

Definition 3. *Let $J : \mathcal{M}(X) \to \overline{\mathbb{R}}$ be a function. Its convex conjugate is a function $J^\star : \mathcal{C}(X) \to \overline{\mathbb{R}}$ defined by*

$$J^\star(\varphi) = \sup_{\mu\in\mathcal{M}(X)} \Big[\int_X \varphi(x)\, \mu(dx) - J(\mu) \Big].$$

Note that J must now be defined on all of $\mathcal{M}(X)$; it is always possible to simply define $J(\mu) = \infty$ if $\mu \notin \mathcal{P}(X)$, although sometimes a different extension may be more convenient. The convex conjugate forms the core of the following representation for the influence function Ψ_μ:

Theorem 2 (Fenchel–Moreau representation). *Let $J : \mathcal{M}(X) \to \overline{\mathbb{R}}$ be proper, convex, and lower semicontinuous. Then the maximizer of $\varphi \mapsto \mathbb{E}_{x\sim\mu}[\varphi(x)] - J^\star(\varphi)$, if it exists, is an influence function for J at μ. With some abuse of notation, we have that*

$$\Psi_\mu = \arg\max_{\varphi\in\mathcal{C}(X)} \Big[\mathbb{E}_{x\sim\mu}[\varphi(x)] - J^\star(\varphi) \Big].$$

Theorem 2 motivates the following influence function approximation strategy: model $\varphi : X \to \mathbb{R}$ with a neural network and train it using stochastic gradient ascent on the objective $\phi \mapsto \mathbb{E}_{x\sim\mu}[\varphi_\phi(x)] - J^\star(\varphi_\phi)$. The trained neural network is then an approximation to Ψ_μ suitable for use in the descent step of PFD. Under this approximation scheme, PFD can be concisely expressed as the saddle-point problem

$$\inf_\mu \sup_\varphi \big[\mathbb{E}_{x\sim\mu}[\varphi(x)] - J^\star(\varphi) \big],$$

where the inner supremum solves for the influence function (the differentiation step of PFD), and the outer infimum descends the linear approximation $\mathbb{E}_{x\sim\mu}[\varphi(x)]$ (the descent step of PFD), noting that $J^\star(\varphi)$ is a constant w.r.t. μ. This procedure is highly reminiscent of adversarial training (Goodfellow et al., 2014); for this reason, we call PFD with this approximation scheme based on convex duality **adversarial PFD**. PFD therefore explains the prevalence of adversarial training as a deep learning technique and extends its applicability to any convex probability functional.

In the following sections, we demonstrate that PFD provides a broad conceptual framework for understanding a wide range of existing machine learning algorithms.

5. Generative Adversarial Networks

Generative adversarial networks (GANs) are a technique to train a parameterized probability measure μ to mimic a data distribution ν. There are many variants of the GAN algorithm. They typically take the form of a saddle-point problem, and it is known that many of them correspond to the minimization of different divergences $D(\mu||\nu)$. We complete the picture by showing that many GAN variants could have been derived as instances of PFD applied to different divergences.

5.1. Minimax GAN

Goodfellow et al. (2014) originally proposed the following saddle-point problem

$$\inf_\mu \sup_D \tfrac{1}{2}\mathbb{E}_{x\sim\nu}[\log D(x)] + \tfrac{1}{2}\mathbb{E}_{x\sim\mu}[\log(1 - D(x))].$$

The interpretation of this *minimax GAN* problem is that the *discriminator* D learns to classify between fake samples from μ and real samples from ν via a binary classification loss, while the *generator* μ is trained to produce counterfeit samples that fool the classifier. It was shown that the value

of the inner optimization problem equals $D_{\mathrm{JS}}(\mu||\nu) - \log 2$, where

$$D_{\mathrm{JS}}(\mu||\nu) = \tfrac{1}{2} D_{\mathrm{KL}}(\mu||\tfrac{1}{2}\mu + \tfrac{1}{2}\nu) + \tfrac{1}{2} D_{\mathrm{KL}}(\nu||\tfrac{1}{2}\mu + \tfrac{1}{2}\nu)$$

is the Jensen–Shannon divergence, and therefore the problem corresponds to training μ to minimize the divergence between μ and ν. As a practical algorithm, simultaneous stochastic gradient descent steps are performed on the discriminator's parameters ϕ and the generator's parameters θ using the two loss functions

$$\begin{cases} \phi \mapsto -\frac{1}{2}\mathbb{E}_{x\sim\nu}[\log D_\phi(x)] - \frac{1}{2}\mathbb{E}_{x\sim\mu_\theta}[\log(1 - D_\phi(x))], \\ \theta \mapsto \frac{1}{2}\mathbb{E}_{x\sim\mu_\theta}[\log(1 - D_\phi(x))], \end{cases} \tag{3}$$

where D_ϕ and μ_θ are parameterized with neural networks.

Our unifying result is the following:

Proposition 1. *Adversarial PFD on the Jensen–Shannon divergence objective*

$$J_{\mathrm{JS}}(\mu) = D_{\mathrm{JS}}(\mu||\nu).$$

yields the minimax GAN algorithm (3)*.*

That is, the minimax GAN could have been derived mechanically and from first principles as an instance of adversarial PFD. To build intuition, we note that the discriminator plays the role of the approximate influence function:

Proposition 2. *Suppose μ has density $p(x)$ and ν has density $q(x)$. Then the influence function for J_{JS} is*

$$\Psi_{\mathrm{JS}}(x) = \frac{1}{2}\log\frac{p(x)}{p(x) + q(x)}.$$

Recall that in the minimax GAN, the optimal discriminator D^* satisfies $D^*(x) = \frac{q(x)}{p(x)+q(x)}$, so the influence function $\Psi_{\mathrm{JS}}(x) = \frac{1}{2}\log(1 - D^*(x))$ is approximated using the learned discriminator.

Now, we rederive the minimax GAN problem (3) as a form of adversarial PFD. We compute:

Proposition 3. *The convex conjugate of J_{JS} is*

$$J^\star_{\mathrm{JS}}(\varphi) = -\tfrac{1}{2}\mathbb{E}_{x\sim\nu}[\log(1 - e^{2\varphi(x)+\log 2})] - \tfrac{1}{2}\log 2.$$

Theorem 2 yields the representation

$$\Psi_{\mathrm{JS}} = \operatorname*{arg\,max}_{\varphi\in\mathcal{C}(X)} \left[\mathbb{E}_{x\sim\mu}[\varphi(x)] + \tfrac{1}{2}\mathbb{E}_{x\sim\nu}[\log(1 - e^{2\varphi(x)+\log 2})]\right],$$

an ascent step on which is the ϕ-step in (3) with the substitution $\varphi = \frac{1}{2}\log(1 - D) - \frac{1}{2}\log 2$. The descent step corresponds to updating μ to decrease the linear approximation $\mathbb{E}_{x\sim\mu}[\varphi(x)]$, which corresponds to the θ-step in (3). In fact, a similar argument can be applied to the f-GANs of Nowozin et al. (2016), which generalize the minimax GAN. The observation that f-GANs (and hence the minimax GAN) can be derived through convex duality was also noted by Farnia & Tse (2018).

5.2. Non-saturating GAN

Goodfellow et al. (2014) also proposed an alternative to (3) called the *non-saturating GAN*, which prescribes descent steps on

$$\begin{cases} \phi \mapsto -\frac{1}{2}\mathbb{E}_{x\sim\nu}[\log D_\phi(x)] - \frac{1}{2}\mathbb{E}_{x\sim\mu_\theta}[\log(1 - D_\phi(x))], \\ \theta \mapsto -\frac{1}{2}\mathbb{E}_{x\sim\mu_\theta}[\log D_\phi(x)]. \end{cases}$$

In the step on the generator's parameters θ, the $\log(1 - D_\phi)$ in the minimax GAN has been replaced with $-\log D_\phi$. This heuristic change prevents gradients to θ from converging to 0 when the discriminator is too confident, and it is for this reason that the loss for θ is called the non-saturating loss.

We consider a slightly modified problem, in which the original minimax loss and the non-saturating loss are summed (and scaled by a factor of 2):

$$\begin{cases} \phi \mapsto -\frac{1}{2}\mathbb{E}_{x\sim\nu}[\log D_\phi(x)] - \frac{1}{2}\mathbb{E}_{x\sim\mu_\theta}[\log(1 - D_\phi(x))], \\ \theta \mapsto -\mathbb{E}_{x\sim\mu_\theta}[\log D_\phi(x)] + \mathbb{E}_{x\sim\mu_\theta}[\log(1 - D_\phi(x))]. \end{cases} \tag{4}$$

This also prevents gradients to θ from saturating, achieving the same goal as the non-saturating GAN. Huszar (2016) and Arjovsky & Bottou (2017) recognize that this process minimizes $D_{\mathrm{KL}}(\mu||\nu)$.[2]

We claim the following:

Proposition 4. *PFD on the reverse Kullback–Liebler divergence objective*

$$J_{\mathrm{NS}}(\mu) = D_{\mathrm{KL}}(\mu||\nu),$$

using the binary classification likelihood ratio estimator to approximate the influence function, yields the modified non-saturating GAN optimization problem (4)*.*

Proposition 5. *Suppose μ has density $p(x)$ and ν has density $q(x)$. The influence function for J_{NS} is*

$$\Psi_{\mathrm{NS}}(x) = \log\frac{p(x)}{q(x)}.$$

Now, because the binary classification loss

$$D \mapsto -\tfrac{1}{2}\mathbb{E}_{x\sim\nu}[\log D(x)] - \tfrac{1}{2}\mathbb{E}_{x\sim\mu_\theta}[\log(1 - D(x))], \tag{5}$$

is minimized by $D(x) = \frac{q(x)}{p(x)+q(x)}$, one estimator for Ψ_{NS} is simply

$$\Psi_{\mathrm{NS}}(x) \approx \log\frac{1 - D_\phi(x)}{D_\phi(x)},$$

[2] The derivation of Huszar (2016) omits showing that the dependence of $\frac{q(x)}{p_\theta(x)}$ on θ can be ignored, but the result is proved by Theorem 2.5 of Arjovsky & Bottou (2017). We remark that this result can be seen as a corollary of Theorem 1 and Proposition 5.

where ϕ is updated as in the ϕ-step in (4). With this approximation scheme, the differentiation step and the descent step in PFD correspond exactly to the ϕ-step and θ-step respectively in (4). Once again, the discriminator serves to approximate the influence function.

5.3. Wasserstein GAN

Arjovsky et al. (2017) propose solving the following saddle-point problem

$$\inf_{\mu} \sup_{||D||_L \leq 1} \big[\mathbb{E}_{x\sim\mu}[D(x)] - \mathbb{E}_{x\sim\nu}[D(x)]\big],$$

where $||D||_L$ denotes the Lipschitz constant of D. The corresponding practical algorithm amounts to simultaneous descent steps on

$$\begin{cases} \phi \mapsto \mathbb{E}_{x\sim\mu_\theta}[D_\phi(x)] - \mathbb{E}_{x\sim\nu}[D_\phi(x)], \\ \theta \mapsto -\mathbb{E}_{x\sim\mu_\theta}[D_\phi(x)], \end{cases} \tag{6}$$

where D_ϕ is reprojected back to the space of 1-Lipschitz functions after each ϕ-step. Here, μ_θ is again the generator, and D_ϕ is the discriminator, sometimes called the critic. This algorithm is called the Wasserstein GAN algorithm, so named because this algorithm approximately minimizes the 1-Wasserstein distance $W_1(\mu, \nu)$; the motivation for the ϕ-step in (6) is so that the discriminator learns the *Kantorovich potential* that describes the optimal transport from μ to ν. See e.g. Villani (2008) for the full optimal transport details.

We claim that the Wasserstein GAN too is an instance of PFD, and once again, the discriminator plays the role of approximate influence function:

Proposition 6. *Adversarial PFD on the Wasserstein distance objective*

$$J_{\mathrm{W}}(\mu) = W_1(\mu, \nu)$$

yields the Wasserstein GAN algorithm (6).

Proposition 7. *The influence function for J_{W} is the Kantorovich potential corresponding to the optimal transport from μ to ν.*

We remark that the gradient computation in Theorem 3 of Arjovsky et al. (2017) is a corollary of Theorem 1 and Proposition 7. Now, we show that the Wasserstein GAN algorithm can be derived mechanically via convex duality. The connection between the Wasserstein GAN and convex duality was also observed by Farnia & Tse (2018).

Proposition 8. *The convex conjugate of J_{W} is*

$$J^\star_{\mathrm{W}}(\varphi) = \mathbb{E}_{x\sim\nu}[\varphi(x)] + \{||\varphi||_L \leq 1\}.$$

We use the notation $\{A\}$ to denote the convex indicator function, which is 0 if A is true and ∞ if A is false.

Theorem 2 yields the representation

$$\Psi_{\mathrm{W}} = \arg\max_{\varphi\in\mathcal{C}(X)} \big[\mathbb{E}_{x\sim\mu_\theta}[\varphi(x)] - \mathbb{E}_{x\sim\nu}[\varphi(x)] - \{||\varphi||_L \leq 1\}\big].$$

The adversarial PFD differentiation step therefore corresponds exactly to the ϕ-step in (6), and the PFD descent step is exactly the θ-step in (6).

6. Variational Inference

In Bayesian inference, the central object is the posterior distribution

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)} = \frac{p(x|z)p(z)}{\int p(x|z)p(z)\,dz},$$

where x is an observed datapoint, $p(x|z)$ is the likelihood, $p(z)$ is the prior. Unfortunately, the posterior is difficult to compute due to the presence of the integral. Variational inference therefore reframes this computation as an optimization problem in which a *variational posterior* $q(z)$ approximates the true posterior by solving

$$\inf_q D_{\mathrm{KL}}(q(z)||p(z|x)).$$

6.1. Black-box variational inference

This objective is not directly optimizable, due to the presence of the intractable $p(z|x)$ term. The tool of choice for variational inference is the *evidence lower bound* (ELBO), which rewrites

$$D_{\mathrm{KL}}(q(z)||p(z|x)) = \log p(x) - \underbrace{\mathbb{E}_{z\sim q(z)}\Big[\log \frac{p(x|z)p(z)}{q(z)}\Big]}_{\text{ELBO}}.$$

Because $\log p(x)$ is fixed, we may maximize the ELBO to minimize the KL divergence. The advantage of doing so is that all the terms inside the expectation are now tractable to evaluate, and thus the expectation may be approximated through Monte Carlo sampling. This leads to the following practical algorithm, namely stochastic gradient descent on the objective

$$\theta \mapsto -\mathbb{E}_{z\sim q_\theta(z)}\Big[\log \frac{p(x|z)p(z)}{q_\theta(z)}\Big]. \tag{7}$$

This is called black-box variational inference (Ranganath et al., 2014). Roeder et al. (2017) later recognized that ignoring the θ-dependence of the term in the expectation yields the same gradients in expectation; it is this variant that we consider. Our unification result is the following:

Proposition 9. *PFD on the variational inference objective*

$$J_{\mathrm{VI}}(q) = D_{\mathrm{KL}}(q(z)||p(z|x)),$$

using exact influence functions, yields the black-box variational inference algorithm (7).

In fact, the influence function turns out to be precisely the inside of the negative ELBO bound:

Proposition 10. *The influence function for J_{VI} is*

$$\Psi_{\mathrm{VI}}(z) = \log \frac{q(z)}{p(x|z)p(z)}.$$

In this context, the influence function can be evaluated exactly, so the differentiation step of PFD may be performed without approximation. The descent step of PFD becomes exactly the descent step on θ of (7), where the θ-dependence of the term in the expectation is ignored. We remark that the argument of Roeder et al. (2017) that this θ-dependence can be ignored can be seen as a corollary of Theorem 1 and Proposition 10.

6.2. Adversarial variational Bayes

When the density function of the prior $p(z)$ or the variational posterior $q(z|x)$ is not available, adversarial variational Bayes (Mescheder et al., 2017) may be employed. Here, the quantity $\log \frac{q(z)}{p(z)}$ is approximated by a neural network $f_\phi(z)$ through a binary classification problem, much like (5). The resulting algorithm applies simultaneous descent steps on

$$\begin{cases} \phi \mapsto -\mathbb{E}_{q_\theta(z)}[\log \sigma(f_\phi(z))] - \mathbb{E}_{p(z)}[\log(1-\sigma(f_\phi(z)))] \\ \theta \mapsto -\mathbb{E}_{q_\theta(z)}[-f_\phi(z) + \log p(x|z)]. \end{cases} \tag{8}$$

This algorithm is another instance of PFD:

Proposition 11. *PFD on the variational inference objective J_{VI}, using the binary classification likelihood ratio estimator to approximate the influence function, yields adversarial variational Bayes* (8)*.*

It is easily seen that

$$\Psi_{\mathrm{VI}}(z) = \log \frac{q(z)}{p(x|z)p(z)} \approx f_\phi(z) - \log p(x|z).$$

Therefore, the ϕ-step of (8) is the differentiation step of PFD, and the θ-step of (8) is the descent step. We remark that the gradient computation in Proposition 2 of Mescheder et al. (2017) is a corollary of Theorem 1 and Proposition 10.

7. Reinforcement Learning

In a Markov decision process, the distribution of states $s = (s_0, s_1, \ldots)$, actions $a = (a_1, a_2, \ldots)$, and rewards $r = (r_1, r_2, \ldots)$ is governed by the distribution

$$\mathbb{P}(s, a, r) = p_0(s_0) \prod_{t=1}^{\infty} p(s_t, r_t|s_{t-1}, a_t)\, \pi(a_t|s_{t-1}),$$

where $p_0(s)$ is an initial distribution over states, $p(s', r|s, a)$ gives the transition probability of arriving at state s' with reward r from a state s taking an action a, and $\pi(a|s)$ is a policy that gives the distribution of actions taken when in state s. In reinforcement learning, we are interested in learning the policy $\pi(a|s)$ that maximizes the expected discounted reward $\mathbb{E}[\sum_{t=1}^{\infty} \gamma^{t-1} r_t]$, where $0 < \gamma < 1$ is a discount factor, while assuming we only have access to samples from p_0 and p.

7.1. Policy iteration

Policy iteration (Howard, 1960; Sutton & Barto, 1998) is one scheme that solves the reinforcement learning problem. It initializes $\pi(s|a)$ arbitrarily and then cycles between two steps, policy evaluation and policy improvement. In the policy evaluation step, the state-action value function $Q^\pi(s, a)$ is computed. In the policy improvement step, the policy is updated to the greedy policy, the policy that at state s takes the action $\arg\max_a Q^\pi(s, a)$ with probability 1.

Before we present our unification result, we introduce an arbitrary distribution over states $\pi(s)$ and consider the joint distribution $\pi(s, a) = \pi(s)\pi(a|s)$, so that π is one probability distribution rather than one for every state s. Now:

Proposition 12. *PFD on the reinforcement learning objective*

$$J_{\mathrm{RL}}(\pi) = -\mathbb{E} \sum_{t=1}^{\infty} \gamma^{t-1} r_t,$$

using exact influence functions and global minimization of the linear approximation, yields the policy iteration algorithm.

Proposition 13. *The influence function for J_{RL} is*

$$\Psi_{\mathrm{RL}}(s, a) = -\frac{\sum_{t=0}^{\infty} \gamma^t p_t^\pi(s)}{\pi(s)} (Q^\pi(s, a) - V^\pi(s)),$$

where Q^π is the state-action value function, V^π is the state value function, and p_t^π is the marginal distribution of states after t steps, all under the policy π.

The descent step of PFD corresponds to taking a step on $\pi_\theta(s, a) = \pi(s)\pi_\theta(a|s)$ to decrease the linear approximation

$$\theta \mapsto -\mathbb{E}_{\pi_\theta(s,a)}\Big[\frac{\sum_{t=0}^{\infty} \gamma^t p_t^\pi(s)}{\pi(s)} (Q^\pi(s, a) - V^\pi(s))\Big].$$

Setting $d^\pi(s) = (1-\gamma)\sum_{t=0}^{\infty} \gamma^t p_t^\pi(s)$, this simplifies to either

$$\theta \mapsto -\frac{1}{1-\gamma} \mathbb{E}_{d^\pi(s)} \mathbb{E}_{\pi_\theta(a|s)} [Q^\pi(s, a) - V^\pi(s)], \tag{9}$$

$$\theta \mapsto -\frac{1}{1-\gamma} \mathbb{E}_{d^\pi(s)} \mathbb{E}_{\pi_\theta(a|s)} [Q^\pi(s, a)] + \text{constant}. \tag{10}$$

The most naive way to decrease (10) is to globally minimize it. This corresponds to setting $\pi_\theta(a|s)$ to be the greedy policy. Hence, the evaluation of $Q^\pi(s,a)$ in policy iteration corresponds exactly to computing the influence function in the differentiation step of PFD, and the greedy policy update corresponds to applying the descent step.

7.2. Policy gradient and actor-critic

Policy iteration exactly computes the linear approximation and nonparametrically minimizes it. Now we consider algorithms in which the policy is parameterized and the descent step is taken using a gradient step on (9) or (10). If this approach is taken, there is a lot of flexibility in how the influence function can be approximated, but generally speaking, the result is an actor-critic method (Konda & Tsitsiklis, 2000; Sutton et al., 2000), which describes a class of algorithms that approximates the value function of the current policy and then takes a gradient step on the parameters of the policy using the estimated value function. We claim:

Proposition 14. *Approximate PFD on the reinforcement learning objective J_{RL}, where the influence function is estimated using, for example, Monte Carlo, least squares, or temporal differences, yields an actor-critic algorithm.*

There is a huge number of possible approximations to the influence function; we list several and their corresponding algorithms. The simplest algorithm is the policy gradient algorithm, also known as REINFORCE (Williams, 1992), which directly uses a Monte Carlo estimate of $Q^\pi(s,a)$ as the influence function estimator. Stochastic value gradients (Heess et al., 2015) and the closely related deterministic policy gradient (Silver et al., 2014) fit a neural network to $Q^\pi(s,a)$ using a temporal difference update and use that as the influence function approximation; their use of a neural network makes them compatible with the reparameterization trick. Advantage actor-critic (Mnih et al., 2016) estimates $Q^\pi(s,a) - V^\pi(s)$ by estimating $Q^\pi(s,a)$ using Monte Carlo and fitting a neural network to $V^\pi(s)$ using least squares. All of these algorithms are traditionally justified by the celebrated policy gradient theorem (Sutton et al., 2000); we remark that this theorem is a corollary of Theorem 1 and Proposition 13.

7.3. Dual actor-critic

Because J_{RL} is not convex, adversarial PFD does not directly apply. However, the form of Proposition 13 strongly suggests fixing the arbitrary distribution $\pi(s)$ to be the discounted marginal distribution of states $d^\pi(s)$. Closely related to the linear programming formulation of reinforcement learning (Puterman, 1994), this choice turns out to convexify J_{RL}, thus enabling the use of convex duality to approximate its influence function. We expect to obtain an adversarial formulation of reinforcement learning; one such formulation is the dual actor-critic algorithm (Dai et al., 2017b; Chen & Wang, 2016):

$$\sup_\pi \inf_V (1-\gamma)\mathbb{E}_{p_0(s)}[V(s)] + \mathbb{E}_{\pi(s,a)}[\mathcal{A}V(s,a)], \quad (11)$$

where $\mathcal{A}V(s,a) = \mathbb{E}_{p(s',r|s,a)}[r+\gamma V(s')] - V(s)$. Indeed:

Proposition 15. *Adversarial PFD on the reinforcement learning objective J_{RL} yields the dual actor-critic algorithm (11).*

Proposition 16. *The convex conjugate of J_{RL} is*

$$J^\star_{\mathrm{RL}}(\varphi) = (1-\gamma)\mathbb{E}_{p_0(s)}V_\varphi(s) + \{V_\varphi \textit{ exists}\},$$

where V_φ is the unique solution to $\varphi = -\mathcal{A}V_\varphi$, if it exists.

Using Theorem 2, adversarial PFD therefore recovers (11):

$$\inf_\pi \sup_\varphi \mathbb{E}_{\pi(s,a)}[\varphi(s,a)] - J^\star_{\mathrm{RL}}(\varphi)$$
$$= \inf_\pi \sup_\varphi \mathbb{E}_{\pi(s,a)}[-\mathcal{A}V_\varphi(s,a)] - (1-\gamma)\mathbb{E}_{p_0(s)}V_\varphi(s).$$

8. Conclusion

This paper suggests several new research directions. First is the transfer of insight and specialized techniques from one domain to another. As just one example, in the context of GANs, Arjovsky et al. (2017) claim that constraining the discriminator to be 1-Lipschitz improves the stability of the training algorithm – could similarly constraining the analogous object in reinforcement learning, namely an approximation to the advantage function, lead to improved stability in deep reinforcement learning?

Moreover, the abstract viewpoint taken in this paper allows for the simultaneous development of new algorithms for GANs, variational inference, and reinforcement learning. General influence function approximation techniques in the spirit of convex duality could improve all three fields at once. More sophisticated descent techniques beyond gradient descent on parameterized probability distributions, such as Frank-Wolfe or trust-region methods, could improve learning or yield valuable convergence guarantees.

Finally, this paper unlocks the possibility of applying probability functional descent to new problems. In principle, the algorithm can be applied mechanically to any situation where one wants to optimize over probability distributions, possibly leading to new, straightforward ways to solve problems in, for example, mathematical finance, mean field games, or POMDPs. One could argue that the current excitement over deep learning began once researchers realized that to solve a problem, they could simply write a loss function and then rely on automatic differentiation and gradient descent to minimize it. We hope that probability functional descent provides a similarly turnkey solution for optimizing loss functions defined on probability distributions and leads to a similar burst of research activity.

Acknowledgements

We thank Rui Shu, Yang Song, Shengjia Zhao, Abubakar Abid, Andrea Zanette, Jordi Feliu Fabà, Jing An, Abeynaya Gnanasekaran, and Kailai Xu for helpful discussions. Support from NSF grants DMS-1720451 and DMS-1820942 is gratefully acknowledged by J. Blanchet.

References

Aliprantis, C. D. and Border, K. C. Infinite dimensional analysis: a hitchhiker's guide. 2006.

Arjovsky, M. and Bottou, L. Towards principled methods for training generative adversarial networks. *arXiv preprint arXiv:1701.04862*, 2017.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein GAN. *arXiv preprint arXiv:1701.07875*, 2017.

Bottou, L. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pp. 177–186. Springer, 2010.

Carmona, R. and Delarue, F. *Probabilistic Theory of Mean Field Games with Applications I-II*. Springer, 2018.

Chen, Y. and Wang, M. Stochastic primal-dual methods and sample complexity of reinforcement learning. *arXiv preprint arXiv:1612.02516*, 2016.

Dai, B. *Learning over Functions, Distributions and Dynamics via Stochastic Optimization*. PhD thesis, Georgia Institute of Technology, 2018.

Dai, B., Xie, B., He, N., Liang, Y., Raj, A., Balcan, M.-F. F., and Song, L. Scalable kernel methods via doubly stochastic gradients. In *Advances in Neural Information Processing Systems*, pp. 3041–3049, 2014.

Dai, B., He, N., Dai, H., and Song, L. Provable Bayesian inference via particle mirror descent. In *Artificial Intelligence and Statistics*, pp. 985–994, 2016.

Dai, B., He, N., Pan, Y., Boots, B., and Song, L. Learning from conditional distributions via dual embeddings. In *Artificial Intelligence and Statistics*, pp. 1458–1467, 2017a.

Dai, B., Shaw, A., He, N., Li, L., and Song, L. Boosting the actor with dual critic. *arXiv preprint arXiv:1712.10282*, 2017b.

Dai, B., Shaw, A., Li, L., Xiao, L., He, N., Liu, Z., Chen, J., and Song, L. SBEED: Convergent reinforcement learning with nonlinear function approximation. In *International Conference on Machine Learning*, pp. 1133–1142, 2018.

Farnia, F. and Tse, D. A convex duality framework for GANs. In *Advances in Neural Information Processing Systems*, pp. 5250–5259, 2018.

Fernholz, L. T. *Von Mises calculus for statistical functionals*, volume 19. Springer Science & Business Media, 2012.

Frogner, C. and Poggio, T. Approximate inference with wasserstein gradient flows. *arXiv preprint arXiv:1806.04542*, 2018.

Gaivoronski, A. Linearization methods for optimization of functionals which depend on probability measures. In *Stochastic Programming 84 Part II*, pp. 157–181. Springer, 1986.

Glynn, P. W. Likelihood ratio gradient estimation for stochastic systems. *Communications of the ACM*, 33 (10):75–84, 1990.

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*, pp. 2672–2680, 2014.

Heess, N., Wayne, G., Silver, D., Lillicrap, T., Erez, T., and Tassa, Y. Learning continuous control policies by stochastic value gradients. In *Advances in Neural Information Processing Systems*, pp. 2944–2952, 2015.

Ho, J. and Ermon, S. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*, pp. 4565–4573, 2016.

Howard, R. A. Dynamic programming and markov processes. 1960.

Huszar, F. An alternative update rule for generative adversarial networks. *Unpublished note (retrieved on 16 Jan 2019)*, 2016.

Kingma, D. P. and Welling, M. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kleijnen, J. P. and Rubinstein, R. Y. Optimization and sensitivity analysis of computer simulation models by the score function method. *European Journal of Operational Research*, 88(3):413–427, 1996.

Konda, V. R. and Tsitsiklis, J. N. Actor-critic algorithms. In *Advances in neural information processing systems*, pp. 1008–1014, 2000.

Lin, A. T., Li, W., Osher, S., and Montúfar, G. Wasserstein proximal of gans. 2018.

Liu, Q. and Wang, D. Stein variational gradient descent: A general purpose Bayesian inference algorithm. In *Advances In Neural Information Processing Systems*, pp. 2378–2386, 2016.

Lucchetti, R. *Convexity and well-posed problems*. Springer Science & Business Media, 2006.

Luenberger, D. G. and Ye, Y. Linear and nonlinear programming. 2015.

Mescheder, L., Nowozin, S., and Geiger, A. Adversarial variational Bayes: Unifying variational autoencoders and generative adversarial networks. *arXiv preprint arXiv:1701.04722*, 2017.

Milgrom, P. and Segal, I. Envelope theorems for arbitrary choice sets. *Econometrica*, 70(2):583–601, 2002.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937, 2016.

Molchanov, I. and Zuyev, S. Variational calculus in the space of measures and optimal design. In *Optimum Design 2000*, pp. 79–90. Springer, 2001.

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

Penot, J.-P. *Calculus without derivatives*, volume 266. Springer Science & Business Media, 2012.

Puterman, M. Markov decision processes: Discrete stochastic dynamic programming. 1994.

Ranganath, R., Gerrish, S., and Blei, D. Black box variational inference. In *Artificial Intelligence and Statistics*, pp. 814–822, 2014.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

Richemond, P. H. and Maginnis, B. On wasserstein reinforcement learning and the fokker-planck equation. *arXiv preprint arXiv:1712.07185*, 2017.

Rockafellar, R. A general correspondence between dual minimax problems and convex programs. *Pacific Journal of Mathematics*, 25(3):597–611, 1968.

Roeder, G., Wu, Y., and Duvenaud, D. K. Sticking the landing: Simple, lower-variance gradient estimators for variational inference. In *Advances in Neural Information Processing Systems*, pp. 6925–6934, 2017.

Ruderman, A., Reid, M., García-García, D., and Petterson, J. Tighter variational representations of f-divergences via restriction to probability measures. *arXiv preprint arXiv:1206.4664*, 2012.

Santambrogio, F. Functionals on the space of probabilities. In *Optimal Transport for Applied Mathematicians*, pp. 249–284. Springer, 2015.

Schulman, J., Heess, N., Weber, T., and Abbeel, P. Gradient estimation using stochastic computation graphs. In *Advances in Neural Information Processing Systems*, pp. 3528–3536, 2015.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. Deterministic policy gradient algorithms. In *ICML*, 2014.

Sutton, R. S. and Barto, A. G. *Introduction to reinforcement learning*, volume 135. MIT Press Cambridge, 1998.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pp. 1057–1063, 2000.

Syed, U., Bowling, M., and Schapire, R. E. Apprenticeship learning using linear programming. In *Proceedings of the 25th international conference on Machine learning*, pp. 1032–1039. ACM, 2008.

Villani, C. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

von Mises, R. On the asymptotic distribution of differentiable statistical functions. *The annals of mathematical statistics*, 18(3):309–348, 1947.

Wang, K.-C., Vicol, P., Lucas, J., Gu, L., Grosse, R., and Zemel, R. Adversarial distillation of Bayesian neural network posteriors. *arXiv preprint arXiv:1806.10317*, 2018.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Zalinescu, C. *Convex analysis in general vector spaces*. World scientific, 2002.

Zhang, R., Chen, C., Li, C., and Carin, L. Policy optimization as wasserstein gradient flows. *arXiv preprint arXiv:1808.03030*, 2018.

Supplementary Material for Probability Functional Descent

A. Proofs and Computations

Lemma 1. *Let $J : \mathcal{P}(X) \to \mathbb{R}$. Then $\Psi : X \to \mathbb{R}$ is an influence function of J at μ if and only if*

$$\frac{d}{d\epsilon} J(\mu + \epsilon\chi)\Big|_{\epsilon=0^+} = \int_X \Psi(x)\, \chi(dx).$$

Proof. The left-hand side equals (1), which equals (2). □

Theorem 1 (Chain rule). *Let $J : \mathcal{P}(X) \to \mathbb{R}$ be continuously differentiable, in the sense that the influence function Ψ_μ exists and $(\mu, \nu) \mapsto \mathbb{E}_{x\sim\nu}[\Psi_\mu(x)]$ is continuous. Let the parameterization $\theta \mapsto \mu_\theta$ be differentiable, in the sense that $\frac{1}{||h||}(\mu_{\theta+h} - \mu_\theta)$ converges to a weak limit as $h \to 0$. Then*

$$\nabla_\theta J(\mu_\theta) = \nabla_\theta \mathbb{E}_{x\sim\mu_\theta}[\hat{\Psi}(x)],$$

where $\hat{\Psi} = \Psi_{\mu_\theta}$ is treated as a function $X \to \mathbb{R}$ that is not dependent on θ.

Proof. Without loss of generality, assume $\theta \in \mathbb{R}$, as the gradient is simply a vector of one-dimensional derivatives. Let $\chi_\epsilon = \frac{1}{\epsilon}(\mu_{\theta+\epsilon} - \mu_\theta)$, and let $\chi = \lim_{\epsilon\to 0} \chi_\epsilon$ (weakly). Then

$$\begin{aligned}\frac{d}{d\theta} J(\mu_\theta) &= \frac{d}{d\epsilon} J(\mu_{\theta+\epsilon})\Big|_{\epsilon=0} \\ &= \frac{d}{d\epsilon} J(\mu_\theta + \epsilon\chi_\epsilon)\Big|_{\epsilon=0}.\end{aligned}$$

Assuming for now that

$$\frac{d}{d\epsilon} J(\mu_\theta + \epsilon\chi_\epsilon)\Big|_{\epsilon=0} = \frac{d}{d\epsilon} J(\mu_\theta + \epsilon\chi)\Big|_{\epsilon=0},$$

we have by Lemma 1 that

$$\begin{aligned}\frac{d}{d\theta} J(\mu_\theta) &= \int_X \hat{\Psi}\, d\chi \\ &= \int_X \hat{\Psi}\, d\Big(\lim_{\epsilon\to 0} \frac{1}{\epsilon}(\mu_{\theta+\epsilon} - \mu_\theta)\Big) \\ &= \lim_{\epsilon\to 0} \int_X \hat{\Psi}\, d\Big(\frac{1}{\epsilon}(\mu_{\theta+\epsilon} - \mu_\theta)\Big) \\ &= \frac{d}{d\theta} \int_X \hat{\Psi}\, d\mu_\theta,\end{aligned}$$

where the interchange of limits is by the definition of weak convergence (recall we assumed that X is compact, so $\hat{\Psi}$ is continuous and bounded by virtue of being continuous).

The equality we assumed is the definition of a stronger notion of differentiability called Hadamard differentiability of J. Our conditions imply Hadamard differentiability via Proposition 2.33 of Penot (2012), noting that the map $(\mu, \chi) \mapsto \int_X \Psi_\mu\, d\chi$ is continuous by assumption. □

Theorem 2 (Fenchel–Moreau representation). *Let $J : \mathcal{M}(X) \to \overline{\mathbb{R}}$ be proper, convex, and lower semicontinuous. Then the maximizer of $\varphi \mapsto \mathbb{E}_{x\sim\mu}[\varphi(x)] - J^\star(\varphi)$, if it exists, is an influence function for J at μ. With some abuse of notation, we have that*

$$\Psi_\mu = \arg\max_{\varphi\in\mathcal{C}(X)} \Big[\mathbb{E}_{x\sim\mu}[\varphi(x)] - J^\star(\varphi)\Big].$$

Proof. We will exploit the Fenchel–Moreau theorem, which applies in the setting of locally convex, Hausdorff topological vector spaces (see e.g. Zalinescu (2002)). The space we consider is $\mathcal{M}(X)$, the space of signed, finite measures equipped with the topology of weak convergence, of which $\mathcal{P}(X)$ is a convex subset. $\mathcal{M}(X)$ is indeed locally convex and Hausdorff, and its dual space is $\mathcal{C}(X)$ (see e.g. Aliprantis & Border (2006), section 5.14).

We now show that a maximizer φ^* is an influence function. By the Fenchel–Moreau theorem,

$$J(\mu) = J^{\star\star}(\mu) = \sup_{\varphi\in\mathcal{C}(X)} \Big[\int_X \varphi\, d\mu - J^\star(\varphi)\Big],$$

and

$$J(\mu + \epsilon\chi) = \sup_{\varphi\in\mathcal{C}(X)} \Big[\int_X \varphi\, d\mu + \epsilon\int_X \varphi\, d\chi - J^\star(\varphi)\Big].$$

Because J is differentiable, $\epsilon \mapsto J(\mu + \epsilon\chi)$ is differentiable, so by the envelope theorem (Milgrom & Segal, 2002),

$$\frac{d}{d\epsilon} J(\mu + \epsilon\chi)\Big|_{\epsilon=0} = \int_X \varphi^*\, d\chi,$$

so that φ^* is an influence function by Lemma 1.

The abuse of notation stems from the fact that not all influence functions are maximizers. This is true, though, if

$J(\mu) = \infty$ if $\mu \notin \mathcal{P}(X)$:

$$\begin{aligned}
&\int_X \Psi_\mu \, d\mu - J^\star(\Psi_\mu) \\
&\quad = \int_X \Psi_\mu \, d\mu - \sup_{\nu \in \mathcal{P}(X)} \Big[\int_X \Psi_\mu \, d\nu - J(\nu) \Big] \\
&\quad = \inf_{\nu \in \mathcal{P}(X)} \Big[- \int_X \Psi_\mu \, d(\nu - \mu) + J(\nu) \Big] \\
&\quad = \inf_{\nu \in \mathcal{P}(X)} \Big[- \frac{d}{d\epsilon} J(\mu + \epsilon(\nu - \mu)) \Big|_{\epsilon = 0} + J(\nu) \Big] \\
&\quad \geq J(\mu),
\end{aligned}$$

since the convex function $f(\epsilon) = J(\mu + \epsilon(\nu - \mu))$ lies above its tangent line:

$$f(1) \geq f(0) + 1 \cdot f'(0).$$

Since $J(\mu) = J^{\star\star}(\mu)$, we have that

$$\int_X \Psi_\mu \, d\mu - J^\star(\Psi_\mu) \geq \sup_{\varphi \in \mathcal{C}(X)} \Big[\int_X \varphi \, d\mu - J^\star(\varphi) \Big].$$

□

The following lemma will come in handy in our computations.

Lemma 2. *Suppose $J : \mathcal{M}(X) \to \overline{\mathbb{R}}$ has a representation*

$$J(\mu) = \sup_{\varphi \in \mathcal{C}(X)} \Big[\int_X \varphi \, d\mu - K(\varphi) \Big],$$

where $K : \mathcal{C}(X) \to \overline{\mathbb{R}}$ is proper, convex, and lower semi-continuous. Then $J^\star = K$.

Proof. By definition of the convex conjugate, $J = K^\star$. Then $J^\star = K^{\star\star} = K$, by the Fenchel–Moreau theorem. □

We note that when applying this lemma, we will often implicitly define the appropriate extension of J to $\mathcal{M}(X)$ to be $J(\mu) = \sup_{\varphi \in \mathcal{C}(X)} [\int \varphi \, d\mu - K(\varphi)]$. The exact choice of extension can certainly affect the exact form of the convex conjugate; see Ruderman et al. (2012) for one example of this phenomenon.

Proposition 2. *Suppose μ has density $p(x)$ and ν has density $q(x)$. Then the influence function for J_{JS} is*

$$\Psi_{\mathrm{JS}}(x) = \frac{1}{2} \log \frac{p(x)}{p(x) + q(x)}.$$

Proof. The result follows from Lemma 1:

$$\begin{aligned}
&\frac{d}{d\epsilon} J_{\mathrm{JS}}(\mu + \epsilon\chi)\Big|_{\epsilon=0} \\
&\quad = \frac{1}{2} \int_X \frac{d}{d\epsilon} \Big[(p + \epsilon\chi) \log \frac{p + \epsilon\chi}{\frac{1}{2}(p + \epsilon\chi) + \frac{1}{2}q} \\
&\qquad\qquad + q \log \frac{q}{\frac{1}{2}(p + \epsilon\chi) + \frac{1}{2}q} \Big]_{\epsilon=0} dx \\
&\quad = \frac{1}{2} \int_X \Big[\log \frac{p}{\frac{1}{2}p + \frac{1}{2}q} + 1 - \frac{p}{p+q} - \frac{q}{p+q} \Big] \chi \, dx \\
&\quad = \frac{1}{2} \int_X \Big[\log \frac{p}{p+q} + \log 2 \Big] \chi \, dx.
\end{aligned}$$

□

Proposition 3. *The convex conjugate of J_{JS} is*

$$J^\star_{\mathrm{JS}}(\varphi) = -\tfrac{1}{2} \mathbb{E}_{x \sim \nu} [\log(1 - e^{2\varphi(x) + \log 2})] - \tfrac{1}{2} \log 2.$$

Proof.

$$\begin{aligned}
J^\star_{\mathrm{JS}}(\varphi) &= \sup_{\mu \in \mathcal{M}(X)} \Big[\int_X \varphi \, d\mu - J_{\mathrm{JS}}(\mu) \Big] \\
&= \sup_p \int_X \Big[\varphi p - \frac{1}{2} p \log \frac{p}{\frac{1}{2}p + \frac{1}{2}q} - \frac{1}{2} q \log \frac{q}{\frac{1}{2}p + \frac{1}{2}q} \Big] dx.
\end{aligned}$$

Setting the integrand's derivative w.r.t. p to 0, we find that pointwise, the optimal p satisfies

$$\varphi = \frac{1}{2} \log \frac{p}{\frac{1}{2}p + \frac{1}{2}q}.$$

We eliminate p in the integrand. Notice that the first two terms in the integrand cancel after plugging in p. Since

$$\frac{q}{\frac{1}{2}p + \frac{1}{2}q} = 2\Big(1 - \frac{p}{p+q}\Big) = 2(1 - 2e^{2\varphi}),$$

we obtain that

$$J^\star_{\mathrm{JS}}(\varphi) = -\frac{1}{2} \int_X q \log(1 - 2e^{2\varphi}) \, dx - \frac{1}{2} \log 2.$$

□

Proposition 5. *Suppose μ has density $p(x)$ and ν has density $q(x)$. The influence function for J_{NS} is*

$$\Psi_{\mathrm{NS}}(x) = \log \frac{p(x)}{q(x)}.$$

Proof. The result follows from Lemma 1:

$$\begin{aligned}
&\frac{d}{d\epsilon} J_{\mathrm{NS}}(\mu + \epsilon\chi)\Big|_{\epsilon=0} \\
&\quad= \frac{d}{d\epsilon} \int_X (p + \epsilon\chi) \log \frac{p + \epsilon\chi}{q}\, dx\Big|_{\epsilon=0} \\
&\quad= \int_X \left[\chi \log \frac{p}{q} + \chi\right] dx \\
&\quad= \int_X \left[\log \frac{p}{q} + 1\right] d\chi \\
&\quad= \int_X \left[\log \frac{p}{q}\right] d\chi.
\end{aligned}$$

□

Proposition 7. *The influence function for J_{W} is the Kantorovich potential corresponding to the optimal transport from μ to ν.*

Proof. See Santambrogio (2015), Proposition 7.17. □

Proposition 8. *The convex conjugate of J_{W} is*

$$J^\star_{\mathrm{W}}(\varphi) = \mathbb{E}_{x\sim\nu}[\varphi(x)] + \{||\varphi||_L \leq 1\}.$$

Proof. Using Kantorovich–Rubinstein duality, we have that

$$\begin{aligned}
J_{\mathrm{W}}(\mu) &= \sup_{||\varphi||_L \leq 1} \left[\int_X \varphi\, d\mu - \int_X \varphi\, d\nu\right] \\
&= \sup_{\varphi} \left[\int_X \varphi\, d\mu - \int_X \varphi\, d\nu - \{||\varphi||_L \leq 1\}\right],
\end{aligned}$$

where we use the notation

$$\{A\} = \begin{cases} 0 & A \text{ is true,} \\ \infty & A \text{ is false.} \end{cases}$$

By Lemma 2,

$$J^\star_{\mathrm{W}}(\varphi) = \int_X \varphi\, d\nu + \{||\varphi||_L \leq 1\}.$$

□

Proposition 10. *The influence function for J_{VI} is*

$$\Psi_{\mathrm{VI}}(z) = \log \frac{q(z)}{p(x|z)p(z)}.$$

Proof. The result follows from Lemma 1:

$$\begin{aligned}
&\frac{d}{d\epsilon} J_{\mathrm{VI}}(q + \epsilon\chi)\Big|_{\epsilon=0} \\
&\quad= \frac{d}{d\epsilon} \int (q(z) + \epsilon\chi(z)) \log \frac{q(z) + \epsilon\chi(z)}{p(z|x)}\, dz\Big|_{\epsilon=0} \\
&\quad= \int \left[\chi(z) \log \frac{q(z) + \epsilon\chi(z)}{p(z|x)} + \chi(z)\right] dz\Big|_{\epsilon=0} \\
&\quad= \int \left[\log \frac{q(z)}{p(z|x)} + 1\right] \chi(z)\, dz \\
&\quad= \int \left[\log \frac{q(z)}{p(x|z)p(z)} + \log p(x) + 1\right] \chi(z)\, dz \\
&\quad= \int \log \frac{q(z)}{p(x|z)p(z)}\, \chi(z)\, dz.
\end{aligned}$$

□

Proofs continue on the following page.

Proposition 13. *The influence function for J_{RL} is*

$$\Psi_{\mathrm{RL}}(s,a) = -\frac{\sum_{t=0}^{\infty}\gamma^t p_t^\pi(s)}{\pi(s)}(Q^\pi(s,a) - V^\pi(s)),$$

where Q^π is the state-action value function, V^π is the state value function, and p_t^π is the marginal distribution of states after t steps, all under the policy π.

Proof. First, we note that

$$\begin{aligned}
&\frac{d}{d\epsilon}(\pi+\epsilon\chi)(a|s)\Big|_{\epsilon=0} \\
&\quad= \frac{d}{d\epsilon}\frac{\pi(a,s)+\epsilon\chi(s,a)}{\pi(s)+\epsilon\chi(s)}\Big|_{\epsilon=0} \\
&\quad= \frac{\chi(s,a)-\chi(s)\pi(a|s)}{\pi(s)},
\end{aligned}$$

where we abuse notation to denote $\chi(s) = \int \chi(s,a')\,da'$.

We have

$$-J_{\mathrm{RL}} = \mathbb{E}\Big[\sum_{t=1}^{\infty}\gamma^{t-1}r_t\Big],$$

or, plugging in the measure,

$$-J_{\mathrm{RL}} = \int \sum_{t=1}^{\infty}\gamma^{t-1}r_t\,p_0(s_0)\prod_{j=1}^{\infty}p(s_j,r_j|s_{j-1},a_j)\prod_{k=1}^{\infty}\pi(a_k|s_{k-1}).$$

The integral is over all free variables; we omit them here and in the following derivation for conciseness.

In computing $\frac{d}{d\epsilon}J_{\mathrm{RL}}(\pi+\epsilon\chi)|_{\epsilon=0}$, the product rule dictates that a term appear for every k, in which $\pi(a_k|s_{k-1})$ is replaced with $\frac{d}{d\epsilon}(\pi+\epsilon\chi)(a_k|s_{k-1})|_{\epsilon=0}$. Hence:

$$\begin{aligned}
&-\frac{d}{d\epsilon}J_{\mathrm{RL}}(\pi+\epsilon\chi)\Big|_{\epsilon=0} \\
&\quad= \int \sum_{t=1}^{\infty}\gamma^{t-1}r_t\,p_0(s_0)\prod_{j=1}^{\infty}p(s_j,r_j|s_{j-1},a_j) \\
&\qquad\times\sum_{k=1}^{\infty}\frac{\chi(s_{k-1},a_k)-\chi(s_{k-1})\pi(a_k|s_{k-1})}{\pi(s_{k-1})}\prod_{\substack{\ell=1\\\ell\neq k}}^{\infty}\pi(a_\ell|s_{\ell-1}) \\
&\quad= \sum_{k=1}^{\infty}\int \sum_{t=1}^{\infty}\gamma^{t-1}r_t\,p_0(s_0)\prod_{j=1}^{\infty}p(s_j,r_j|s_{j-1},a_j) \\
&\qquad\times\frac{\chi(s_{k-1},a_k)-\chi(s_{k-1})\pi(a_k|s_{k-1})}{\pi(s_{k-1})}\prod_{\substack{\ell=1\\\ell\neq k}}^{\infty}\pi(a_\ell|s_{\ell-1}),
\end{aligned}$$

reordering the summations. Note that for $t < k$, the summand vanishes:

$$\begin{aligned}
&\int \prod_{j=k}^{\infty} p(s_j, r_j | s_{j-1}, a_j) \\
&\qquad \times \big(\chi(s_{k-1}, a_k) - \chi(s_{k-1})\pi(a_k|s_{k-1})\big) \prod_{\ell=k+1}^{\infty} \pi(a_\ell|s_{\ell-1}) \\
&= \int \big(\chi(s_{k-1}, a_k) - \chi(s_{k-1})\pi(a_k|s_{k-1})\big) \\
&= \int \big(\chi(s_{k-1}) - \chi(s_{k-1})\big) \\
&= 0,
\end{aligned}$$

since all the variables $a_k, r_k, s_k, a_{k+1}, r_{k+1}, s_{k+1}, \ldots$ integrate away to 1. This yields:

$$\begin{aligned}
&-\frac{d}{d\epsilon} J_{\mathrm{RL}}(\pi + \epsilon\chi)\Big|_{\epsilon=0} \\
&\quad = \sum_{k=1}^{\infty} \int \sum_{t=k}^{\infty} \gamma^{t-1} r_t \, p_0(s_0) \prod_{j=1}^{\infty} p(s_j, r_j | s_{j-1}, a_j) \\
&\qquad \times \frac{\chi(s_{k-1}, a_k) - \chi(s_{k-1})\pi(a_k|s_{k-1})}{\pi(s_{k-1})} \prod_{\substack{\ell=1 \\ \ell \neq k}}^{\infty} \pi(a_\ell|s_{\ell-1}).
\end{aligned}$$

Then, substituting the marginal distribution (note s_{k-1} is not integrated)

$$p^{\pi}_{k-1}(s_{k-1}) = \int \prod_{j=1}^{k-1} p(s_j, r_j | s_{j-1}, a_j) \prod_{\ell=1}^{k-1} \pi(a_\ell|s_{\ell-1}),$$

we obtain

$$\begin{aligned}
&-\frac{d}{d\epsilon} J_{\mathrm{RL}}(\pi + \epsilon\chi)\Big|_{\epsilon=0} \\
&\quad = \sum_{k=1}^{\infty} \int \sum_{t=k}^{\infty} \gamma^{t-1} r_t \, p^{\pi}_{k-1}(s_{k-1}) \prod_{j=k}^{\infty} p(s_j, r_j | s_{j-1}, a_j) \\
&\qquad \times \frac{\chi(s_{k-1}, a_k) - \chi(s_{k-1})\pi(a_k|s_{k-1})}{\pi(s_{k-1})} \prod_{\ell=k+1}^{\infty} \pi(a_\ell|s_{\ell-1}).
\end{aligned}$$

Let us rename the integration variables by decreasing their indices by $k-1$:

$$\begin{aligned}
&-\frac{d}{d\epsilon} J_{\mathrm{RL}}(\pi + \epsilon\chi)\Big|_{\epsilon=0} \\
&\quad = \sum_{k=1}^{\infty} \int \sum_{t=1}^{\infty} \gamma^{t+k-2} r_t \, p^{\pi}_{k-1}(s_0) \prod_{j=1}^{\infty} p(s_j, r_j | s_{j-1}, a_j) \\
&\qquad \times \frac{\chi(s_0, a_1) - \chi(s_0)\pi(a_1|s_0)}{\pi(s_0)} \prod_{\ell=2}^{\infty} \pi(a_\ell|s_{\ell-1}).
\end{aligned}$$

Substituting in

$$\begin{aligned}
V^{\pi}(s_0) &= \int \sum_{t=1}^{\infty} \gamma^{t-1} r_t \prod_{j=1}^{\infty} p(s_j, r_j | s_{j-1}, a_j) \prod_{\ell=1}^{\infty} \pi(a_\ell|s_{\ell-1}), \\
Q^{\pi}(s_0, a_1) &= \int \sum_{t=1}^{\infty} \gamma^{t-1} r_t \prod_{j=1}^{\infty} p(s_j, r_j | s_{j-1}, a_j) \prod_{\ell=2}^{\infty} \pi(a_\ell|s_{\ell-1}),
\end{aligned}$$

we obtain

$$-\frac{d}{d\epsilon}J_{\mathrm{RL}}(\pi+\epsilon\chi)\Big|_{\epsilon=0} = \sum_{k=1}^{\infty}\int \gamma^{k-1}\, p_{k-1}^{\pi}(s_0)\frac{Q^{\pi}(s_0,a_1)\chi(s_0,a_1) - V^{\pi}(s_0)\chi(s_0)}{\pi(s_0)}.$$

Finally, by Lemma 1, we obtain that

$$\Psi_{\mathrm{RL}}(s,a) = -\frac{\sum_{k=0}^{\infty}\gamma^k p_k^{\pi}(s)}{\pi(s)}(Q^{\pi}(s,a) - V^{\pi}(s)).$$

□

Proposition 16. *The convex conjugate of J_{RL} is*

$$J_{\mathrm{RL}}^{\star}(\varphi) = (1-\gamma)\mathbb{E}_{p_0(s)}V_{\varphi}(s) + \{V_{\varphi}\ \textit{exists}\},$$

where V_{φ} is the unique solution to $\varphi = -\mathcal{A}V_{\varphi}$, if it exists.

Proof. As mentioned in the text, we set the arbitrary distribution $\pi(s) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t p_t^{\pi}(s)$. In doing so, $\pi(s,a)$ becomes a state-action *occupancy measure* that describes the frequency of encounters of the state-action pair (s,a) over trajectories governed by the policy $\pi(a|s)$. It is known that there is a bijection between occupancy measures $\pi(s,a)$ and policies $\pi(a|s)$ (Syed et al., 2008; Ho & Ermon, 2016).

We can enforce this setting by redefining

$$J_{\mathrm{RL}}(\pi) = -\mathbb{E}\sum_{t=1}^{\infty}\gamma^{t-1}r_t + \Big\{\forall s : \pi(s) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t p_t^{\pi}(s)\Big\},$$

where again $\{\cdot\}$ is the convex indicator function. This equation can be rewritten as

$$J_{\mathrm{RL}}(\pi) = -\mathbb{E}_{\pi(s,a)}R(s,a) + \Big\{\forall s' :\ \pi(s') = (1-\gamma)p_0(s') + \gamma\mathbb{E}_{\pi(s,a)}p(s'|s,a)\Big\},$$

where $R(s,a) = \mathbb{E}_{p(s',r|s,a)}[r]$. The constraint is known as the *Bellman flow equation*. This formulation is convex, as it is the sum of an affine function and an indicator of a convex set (indeed, an affine subspace).

We recall $-\varphi = \mathcal{A}V_{\varphi}$, where $\mathcal{A}V(s,a) = \mathbb{E}_{p(s',r|s,a)}[r+\gamma V(s')] - V(s)$. Now, V_{φ} is uniquely defined by φ if a solution to the equation exists. To see this, note that V_{φ} is the fixed point of the Bellman operator $\mathcal{T}^a$ defined by

$$(\mathcal{T}^a V)(s) = (R+\varphi)(s,a) + \gamma\mathbb{E}_{p(s'|s,a)}V(s'),$$

which is contractive and therefore has a unique fixed point. A representation of V_{φ} may be obtained via fixed point iteration using $\mathcal{T}^a$ for an arbitrary action a:

$$V_{\varphi}(s) = \lim_{k\to\infty}(\mathcal{T}^a)^k 0 = \mathbb{E}^a\sum_{t=1}^{\infty}\gamma^{t-1}(R+\varphi)(s_t,a),$$

where the expectation is taken under the deterministic policy a.

We rewrite J_{RL} using a Lagrange multiplier $V(s)$

$$\begin{aligned} J_{\mathrm{RL}}(\pi) &= -\mathbb{E}_{\pi(s,a)}R(s,a) + \sup_{V}\int V(s')\Big[\pi(s') - (1-\gamma)p_0(s') - \gamma\mathbb{E}_{\pi(s,a)}p(s'|s,a)\Big]\,ds' \\ &= \sup_{V} -\mathbb{E}_{\pi(s,a)}R(s,a) + \mathbb{E}_{\pi(s)}V(s) - (1-\gamma)\mathbb{E}_{p_0(s)}V(s) - \gamma\mathbb{E}_{\pi(s,a)}\mathbb{E}_{p(s'|s,a)}V(s') \\ &= \sup_{\varphi}\mathbb{E}_{\pi(s,a)}\varphi(s,a) - (1-\gamma)\mathbb{E}_{p_0(s)}V_{\varphi}(s) - \{V_{\varphi}\ \text{exists}\}. \end{aligned}$$

Note that $(1-\gamma)\mathbb{E}_{p_0(s)}V_\varphi(s) + \{V_\varphi \text{ exists}\}$ is convex in φ; this stems from the fact that

$$V_{\alpha\varphi+(1-\alpha)\varphi'} = \alpha V_\varphi + (1-\alpha)V_{\varphi'}.$$

The result follows from Lemma 2.

□