A Closer Look at Gradient Estimators with Reinforcement Learning as Inference

J. Wilder Lavington
Department of Computer Science
University of British Columbia
Vancouver, BC V6T 1Z4, Canada
jola2372@cs.ubc.ca

Michael Teng
Department of Engineering Science
University of Oxford
Oxford OX1 2JD, UK
mteng@robots.ox.ac.uk

Mark Schmidt
Department of Computer Science
University of British Columbia
Vancouver, BC V6T 1Z4, Canada
schmidtm@cs.ubc.ca

Frank Wood
Department of Computer Science
University of British Columbia
Vancouver, BC V6T 1Z4, Canada
fwood@cs.ubc.ca

Abstract

The concept of reinforcement learning as inference (RLAI) has led to the creation of a variety of popular algorithms in deep reinforcement learning. Unfortunately, most research in this area relies on wider algorithmic innovations not necessarily relevant to such frameworks. Additionally, many seemingly unimportant modifications made to these algorithms, actually produce inconsistencies with the original inference problem posed by RLAI. Taking a divergence minimization perspective, this work considers some of the practical merits and theoretical issues created by the choice of loss function minimized in the policy update for off-policy reinforcement learning. Our results show that while the choice of divergence rarely has a major affect on the sample efficiency of the algorithm, it can have important practical repercussions on ease of implementation, computational efficiency, and restrictions to the distribution over actions.

1 Introduction

The recent success of deep reinforcement learning (RL) across a wide range of applications is largely attributable to advances in actor-critic algorithms [25, 19, 22, 37, 49, 40, 50, 32]. While versatile, many of these algorithms are difficult to implement and require careful hyper-parameter tuning [15]. Additionally, the most performant implementations of these algorithms rely on heuristics to stabilize learning that are not explicitly referenced within the algorithm definition, or even informed by the relevant theory [15, 54]. In some cases these implementation details become so crucial to performance that reproducing the original results becomes almost impossible [30, 14, 26].

In light of these issues, some researchers refocused on algorithms informed by probabilistic methods, most notably a popular class of algorithms derived within the 'RL as Inference' (RLAI) framework [43, 35, 34, 59, 44, 17, 2, 51, 8, 47]. Under this framework, policy learning in a Markov decision process can be framed as an approximate posterior inference task, which has several key benefits over alternative approaches. First, policies trained using this approach can produce agents that transfer skills between tasks or even entire environments [16]. Second, RLAI couches the task of model learning (i.e. learning the dynamics of the environment) within the larger literature of generative modeling and can take advantage of existing tools and techniques [10, 2, 28, 8], allowing us to reason in the even wider class of differentiable simulation [39, 18, 3]. Lastly, Kullback–Leibler (KL) regularization introduces the notion of "default policies", which induce different behaviors in the learned policy that are useful in solving certain environments and impriving exploration [20, 24, 23].

Practically speaking, RLAI provides insights into how to construct various policy losses which interact more intelligently with modern computation graph software [45, 1, 6]. In framing RL as an approximate inference task, we force this consideration in deciding what divergence we wish to minimize, and how gradients of that expectation will be estimated. This is doubly important in settings where the the simulator itself can be fully or partially differentiable [6, 5]. While we will not consider how gradient estimates of the forward and backwards messages can be better estimated in general, we will look at how to provide learning signal to policies in a general setting (e.g. for anonymous policy distributions given only log-probability of samples). We argue that closer consideration of this class of gradient estimators will allow for more composable reinforcement learning software, which in turn would allow for more stable policy learning over all model or distribution classes.

This paper focuses on the following: (1) gradient estimators of the forward and reverse KL divergences with respect to the policy parameters, (2) the practical implementations of popular algorithms based on these estimators, and (3) an empirical comparison of the two resulting policy losses. The paper is organized as follows, Section 2 reviews RL and RLAI notation and background, sections 3 and 4 derive policy gradient estimators based on divergence minimization, and section 5 considers a simple experiment to show how the performance of each objective differs under modified hyper-parameters. We conclude with a discussion on the utility of each policy loss, and consider future directions.

2 Background

This section details three important ideas required to understand the differences between gradient estimators and target objectives in RLAI. The first is the probabilistic structure of a Markov Decision process, which will allow us to construct a corresponding inference task and model dependencies within various gradient estimators. The second is the is how actor-critic algorithms amortize many of the previous costly sampling operations required in classical inference based methods and on-policy algorithms. Lastly, we need a basic understanding of the RLAI inference task, and how it can be re-framed as divergence minimization.

2.1 Markov Decision Processes

We define a Markov decision process (MDP), $\mathcal{M}_\Phi(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{T}_0, \mathcal{T}, \Pi_\Phi)$, as a random process which produces a sequence of tuples $\tau_t := \{a_t, s_t, s_{t+1}, r_t\}$, for a particular set of states $s_t \in \mathcal{S}$, actions $a_t \in \mathcal{A}$, initial state distribution $p(s_0) \in \mathcal{T}_0$, transition distribution $p(s_{t+1}|s_t, a_t) \in \mathcal{T}$, reward function $r_t : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and policy $\pi_\phi \in \Pi_\phi : S \to A$ parameterized by ϕ. The generative model for this finite horizon process is defined:

$$q_{\pi_\phi}(\tau) = p(s_0) \prod\nolimits_{t=0}^{T} p(s_{t+1}|s_t, a_t)\pi_\phi(a_t|s_t). \tag{1}$$

Here we also denote the marginal distribution with respect to a state s, under the trajectory distribution given in equation 1, as $q_{\pi_\phi}(s)$. In this setting, RL seeks to recover the policy which maximizes the expected cumulative reward over a trajectory, $\phi^* = \arg\max_{\phi\in\Phi} \mathbb{E}_{q_{\pi_\phi}(\tau)}[\sum_{t=0}^{T} r_t(s_t, a_t)]$. More generally, we can extend this notion to infinite horizon problems through the use of a discount factor which signifies the non-zero probability of termination after every time-step. In this setting it is common to arrive at a policy through the solution to a fixed point equation [4, 53], which can be rewritten as the following optimization problem with respect to the policy parameters ϕ:

$$\phi^* = \max_{\phi\in\Phi} \mathbb{E}_{s\sim d^{\pi_\phi}, a\sim\pi_\phi}[Q^{\pi_\phi}(a, s)], \quad \text{where} \quad d^{\pi_\phi}(s) = (1-\gamma)\sum_{t=0}^{\infty} \gamma^t q_{\pi_\phi}(s_t = s), \tag{2}$$

$$\text{and} \quad Q^{\pi_\phi}(a, s) = \mathbb{E}_{p(s'|s,a)}\left[r(s, a) + \gamma \mathbb{E}_{\pi_\phi(a'|s')}\left[Q^{\pi_\phi}(a', s')\right]\right]. \tag{3}$$

Here, the distribution over states, $d^{\pi_\phi}(s)$, is referred to as the *state occupancy*. Methods which simultaneously learn the policy π_ϕ and critic Q^{π_ϕ} are generally called actor-critic algorithms [40, 25, 50, 48], and represent the most popular class of reinforcement learning algorithms currently available.

2.2 Soft Actor Critic

One popular variant of this framework is known as soft actor-critic (SAC). This algorithm is similar to most off-policy actor critic algorithms, in that it learns to estimate the expected reward ahead given the state and action (the Q-function), using the bellman error:

$$F(\theta) = \mathop{\mathbb{E}}_{(a_t, s_t, s_{t+1}) \sim \mathcal{D}} \left[\left(Q_\theta^{\pi_\phi}(a_t, s_t) - \left(\hat{r}_t + \mathop{\mathbb{E}}_{a_{t+1} \sim \pi_\phi(a_{t+1}|s_{t+1})} \left[\gamma \bar{Q}_\theta^{\pi_\phi}(a_{t+1}, s_{t+1}) \right] \right) \right)^2 \right] \tag{4}$$

Here, D defines the set of examples gathered from the environment under a behavioral policy, or a sequence of behavioral policies. The primary distinction between SAC and other AC algorithms [40, 19] however, is that r_t is replaced with a surrogate $\hat{r_t} = r_t - \mathcal{H}_t$, where $\mathcal{H}_t$ indicates the policy entropy at state s_t. Additionally in this setting, the bar over Q indicates a target network, generally defined by the use of parameter averaging [41, 57], a model ensemble, or both [27]. Using this Q function estimate, we can iteratively update our policy to match the so called "softmax-optimal" policy [24, 25, 35]. This approach defines its policy objective based on a divergence between the current policy and a normalized target Q function:

$$J_\theta(\phi) = \mathop{\mathbb{E}}_{s_t \sim \mathcal{D}} \left[\mathop{\mathbb{E}}_{a_t \sim \pi_\phi(a_t|s_t)} \left[Q_\theta^{\pi_\phi}(a_t, s_t) + \alpha \log \left(\frac{\pi_0(a_t|s_t)}{\pi_\phi(a_t|s_t)} \right) \right] \right], \tag{5}$$

Here the gradient is evaluated using the reparametrization trick [33] in cases where the distribution over actions is reparameterizable (e.g. normal, gamma ect.), and with REINFORCE [58] otherwise. A description of SAC is included in Algorithm 1. To ensure a fair comparison of the two divergences described within this paper, we use SAC as the base algorithm, and modify only the policy update with the corresponding gradient estimator under consideration (colored blue in Algorithm 1).

Algorithm 1 Soft Actor Critic with Entropy Tuning

```
Initialize Q-functions, $Q_{\theta_1}(s,a)$, $Q_{\theta_2}(s,a)$ and policy weights $\pi_\phi(a_t|s_t)$
Initialize target networks $Q_{\bar{\theta}_1}(s,a)$ , $Q_{\bar{\theta}_2}(s,a)$
Initialize replay buffer $D$
for each iteration do
    for each environment step do
        Sample action form policy $\pi$
        store transition into replay buffer
    end for
    for each gradient step do
        Update Q-function using $J_Q(\theta_i)$ for $i \in \{1,2\}$ $\theta_i \leftarrow \theta_i - \lambda_Q \hat{\nabla}_{\theta_i} J_Q(\theta_i)$
        Update policy weights using $J_\pi(\phi)$ $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$
        Adjust temperature using $J(\alpha)$ $\alpha \leftarrow \alpha - \lambda \hat{\nabla}_\alpha J(\alpha)$
        Update target network weights using Polyak averaging $\bar{\theta}_i \leftarrow \tau\theta_i + (1-\tau)\bar{\theta}_i$
    end for
end for
```

2.3 Reinforcement Learning as Inference

RL can be posed as a posterior inference problem over the set of trajectories conditioned on how well we would like the agent to do. This breaks down into a set of latent random variables (the trajectories), and a set of observed random variables describing how well the agent is performing in the classical sense (e.g. the reward). We will refer to the observed random variables as "optimality" variables, denoted O_t. These optimality variables indicate if a given state action pair was optimal or not, and are assumed to be Bernoulli distributed. We say that a state action was optimal by indicating $O_t = 1$, while we say that a state action was sub-optimal by indicating $O_t = 0$. In RLAI, O_t is distributed as,

$$p(O_t = 0|a_t, s_t) = 1 - \frac{1}{Z_0} \exp r(a_t, s_t), \quad \text{and} \quad p(O_t = 1|a_t, s_t) = \frac{1}{Z_0} \exp r(a_t, s_t). \tag{6}$$

We note that Z_0 is dependent on the range of values the reward can take on. In general it is assumed that the reward is strictly negative, making a simple choice for $Z_0 = 1$. With optimality defined, we can consider the probability that a given trajectory, a sequence of state action pairs sampled from a policy interacting with the environment, is itself optimal. For finite horizon problems, this joint distribution over both latent and observed random variables is defined,

$$p(\tau, O_{1:T-1}) = p(s_1) \prod_{t=1}^{T-1} \Big[p(s_{t+1}|s_t, a_t) \pi_0(a_t|s_t) p(O_t|\tau_t) \Big], \tag{7}$$

where $\pi_0(a_t|s_t)$ denotes the 'default' policy or prior over actions given states. For clarity, probabilistic graphical models with different reward and state assumptions are given in Figure 1. However in control we don't just want a metric on how likely a given trajectory is to be optimal, we want to produce a policy which itself induces trajectories that are optimal: $\pi^*(a_t|s_t) = \pi(a_t|s_t, o_t, ..., o_{T-1})$. Unfortunately, arriving at an exact $\pi^*(a_t|s_t)$ is intractable in all but the simplest cases. Whats more, both our graphical model and our resulting joint distribution are defined in terms of entire trajectories. This means that not only must we perform a posterior inference task (e.g. determine what the distribution over states and actions given optimality), but we must also find a way recover the marginal distribution over actions given the state and optimality for any state which we might see through environment interaction. We address the first issue (intractability of the posterior inference problem) by using variational inference to produce an approximate posterior efficiently. We will then show through manipulation of the expected gradient and structure of the graphical model, we can define an objective who's optimum characterizes the same policy as is defined by $\pi(a_t|s_t, o_t, ..., o_{T-1})$.

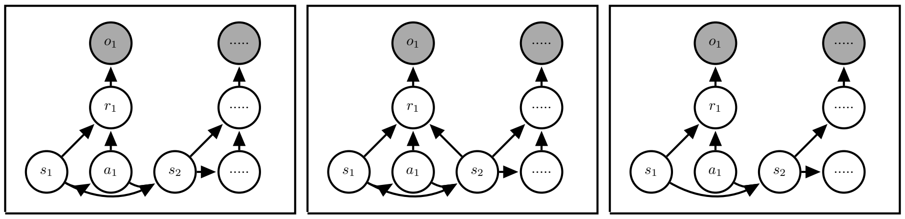

Figure 1: Three potential probabilistic graphical models associated with RLAI. In the graphs, shaded nodes indicate observed random variables, while unobserved or *latent* random variables are transparent. Note that on the left, as is usually the case the reward is not dependent on the resulting transition. The center graphical model gives an example of this, and is often not used as it tends to exacerbate over-optimistic behavior in the policy. The case on the right is given in the "maximum entropy" setting where the default policy is assumed to be uniform and independent of the state.

3 Policy Gradient Estimators Via The Reverse KL Divergence

This section contains derivations relevant to the reverse KL applied to RLAI. More specifically, this section gives information on both the un-marginalized and marginalized objective and its associated gradient estimator. This section will rely on classic tools found in variational inference, and makes frequent use of the REINFORCE trick [58], and the score function trick [52].

3.0.1 The Un-marginalized Reverse KL Objective

For the following we assume that $\tau_t := (a_t, s_t, s_{t+1}, r_t)$, and $(s_{0:T}, a_{0:T-1}, r_{0:T-1}) := \tau$. We begin by looking at the objective derived by integrating over all time-steps simultaneously and differentiating. To derive this objective we start with the functional form of the reverse KL divergence,

$$KL(q_\phi(\tau)||p(\tau|O)) = \int q_\phi(\tau) \log \left(\frac{q_\phi(\tau)}{p(\tau|O)} \right) d\tau$$

$$= \int q_\phi(\tau) \log \left(\frac{q_\phi(\tau)}{p(\tau, O)} \right) d\tau - \int q_\phi(\tau) \log p(O) d\tau = \int q_\phi(\tau) \log \left(\frac{q_\phi(\tau)}{p(\tau, O)} \right) d\tau - \log p(O)$$

Next we can look at the log term and note, that it can be broken up into each time step, and additionally that the dynamics between the the joint and approximate distribution cancel out. We can then simplify the integral to exclude terms which are constant with respect to the policy parameters. This allows us to convert a difficult to compute objective which included the optimal policy, into an objective we can estimate through Monte-Carlo sampling and interaction with the environment:

$$\min_{\phi}\left[\int_{\tau} q_{\phi}(\tau)\log\left(\frac{q_{\phi}(\tau)}{p(\tau,O)}\right)d\tau\right] = \min_{\phi} - \mathop{\mathbb{E}}_{\tau\sim q_{\phi}}\left[\sum_{t=0}^{T-1} r(s_t,a_t) - \log\left(\frac{\pi_0(a_t|s_t)}{\pi_{\phi}(a_t|s_t)}\right)\right].$$

3.0.2 A Marginalized Reverse KL Gradient Estimator

Unfortunately, the naive gradient estimator using REINFORCE on this loss will have variance which grows with the time-horizon T. However by considering its gradient estimator we can infer an objective whose minimum is the same as the one above up to a constant factor. In a similar fashion to the policy gradient theorem [53] we consider the expected gradient of our loss and simplify:

$$\nabla_{\phi}\mathop{\mathbb{E}}_{q(\tau)}\left[\sum_{t=0}^{T-1} r(s_t,a_t)-\log\left(\frac{\pi_{\phi}(a_{t'}|s_{t'})}{\pi_0(a_{t'}|s_{t'})}\right)\right]=\sum_{t=0}^{T-1}\mathop{\mathbb{E}}_{q_{\phi}(\tau_{t:T-1})}\left[\hat{\nabla}_{\phi}\left(\sum_{t'=t}^{T} r(s_{t'},a_{t'})-\log\frac{\pi_{\phi}(a_{t'}|s_{t'})}{\pi_0(a_{t'}|s_{t'})}\right)\right],$$

where $\hat{\nabla}_{\phi} = \nabla_{\phi}\log\pi_{\phi}(s_t,a_t)$. We start by passing the gradient into the expectation under the assumption that the integrand is bounded over the domain of interest,

$$\begin{aligned}\nabla_{\phi}J_{q,p}(\phi) &= \int\nabla_{\phi}\left[\left(\sum_{t=0}^{T-1} r(s_t,a_t)-\log\frac{\pi_{\phi}(a_t|s_t)}{\pi_0(a_t|s_t)}\right)q_{\phi}(\tau)\right]d\tau\\ &= \int\left(\sum_{t=0}^{T-1}\nabla_{\phi}\log\pi_{\phi}(a_t|s_t)\right)\left(\sum_{t'=0}^{T-1} r(s_{t'},a_{t'})-\log\frac{\pi_{\phi}(a_{t'}|s_{t'})}{\pi_0(a_{t'}|s_{t'})}-1\right)q_{\phi}(\tau)d\tau\end{aligned}$$

Note that s_t and s'_t correspond to the *same* random variable (as do a_t and a'_t for all t). The primes are introduced only to index the double sum below correctly. Here we use the score function trick to transform the integral of a gradient into the expectation over a gradient. We then combine terms and simplify, noting that the gradient of the dynamics is zero. We then convolve these sums and further simplify using d separation of the probabilistic graphical model.

$$\begin{aligned}J_{q,p}(\phi) &= \int\sum_{t=0}^{T-1}\sum_{t'=0}^{T-1}\nabla_{\phi}\log\pi_{\phi}(a_t|s_t)\left(r(s_{t'},a_{t'})-\log\frac{\pi_{\phi}(a_{t'}|s_{t'})}{\pi_0(a_{t'}|s_{t'})}-1\right)q_{\phi}(\tau)d\tau,\\ &= \sum_{t=0}^{T-1}\sum_{t'=t}^{T-1}\int\nabla_{\phi}\log\pi_{\phi}(a_t|s_t)\left(r(s_{t'},a_{t'})-\log\frac{\pi_{\phi}(a_{t'}|s_{t'})}{\pi_0(a_{t'}|s_{t'})}\right)q_{\phi}(\tau)d\tau.\end{aligned}$$

Where the last line follows from the Markov property of the graphical model which gives $t' < t \rightarrow r(s_{t'},a_{t'}) - \log\pi_{\phi}(a_{t'}|s_{t'}) + \log\pi_0(a_{t'}|s_{t'})$ is independent of $\nabla_{\phi}\log\pi_{\phi}(a_t|s_t)$. This means that we can decompose the expectation such that for all $t' < t$,

$$\int\nabla_{\phi}\log\pi_{\phi}(a_t|s_t)f(a_{t'},s_{t'})q_{\phi}(\tau)d\tau = \int f(a_{t'},s_{t'})\,(0)\,\frac{q_{\phi}(\tau)}{\pi_{\phi}(a_t|s_t)}d\tau \setminus a_t = 0$$

Therefore the inner expectation is a constant for all t' less then t, and the score function estimator is zero in expectation at these terms in the series. This sum of (entropy regularized) rewards ahead is often referred to as the Q function or the advantage (if we include a baseline or value function which integrates over all actions). This term can be defined more explicitly as,

$$Q(s_t,a_t) = \mathop{\mathbb{E}}_{q_{\phi}(\tau_{t+1:T}|a_t,s_t)}\left[\sum_{t'=t}^{T} r(s_{t'},a_{t'})-\log\frac{\pi_{\phi}(a_{t'}|s_{t'})}{\pi_0(a_{t'}|s_{t'})}\right].$$

Using this estimate of the expected rewards ahead, we can re-scale the log probability of certain actions and give them more weight in the gradient estimator. Crucially, the variance of this gradient estimator is strictly lower, while maintaining the correct target distribution of interest. The full marginalized objective and its estimator are given as the following:

$$\nabla_{\phi}J_{q,p}(\phi)=\sum_{t=0}^{T-1}\mathop{\mathbb{E}}_{q_{\phi}(a_t,s_t)}[\nabla_{\phi}\log\pi_{\phi}(a_t|s_t)Q(s_t,a_t)]\approx\frac{1}{NM}\sum_{t=0}^{T}\sum_{m,n}[\nabla_{\phi}\log\pi_{\phi}(a_t^m|s_t^n)Q(s_t^n,a_t^m)],$$

$$\text{where}\quad a_t^m\sim\pi_{\phi}(a_t|s_t^n),\quad\text{and}\quad s_t^n\sim q_{\phi}(s_t).$$

4 Reweighted Policy Gradient Estimators Via the Forward KL

This section details derivations for how to produce an approximately optimal policy under the forward KL, and like the previous section we will produce a lower variance gradient estimator which targets the original distribution of interest. Here we take advantage of tools from self normalized importance weighting to produce a biased but consistent estimator, and briefly discuss practical issues.

4.0.1 Un-marginalized Objective Derivation for the Forward KL

This objective, represents the expected KL divergence between a target distribution, and an approximate distribution under samples from the target distribution given by $KL(p(\tau|O)||q_\phi(\tau))$. Again, our latent variables are given as the associated state-action pairs, while the observed variables represents the optimality distribution. In order to evaluate the expression above, we apply a standard importance weighting scheme, beginning with the removal constant terms within the expectation.

$$\begin{aligned} KL(p(\tau|O)||q_\phi(\tau)) &= \int p(\tau|O)\log\frac{p(\tau|O)}{q_\phi(\tau)}d\tau = -\int p(\tau|O)\log q_\phi(\tau)d\tau + \int p(\tau|O)\log p(\tau|O)d\tau \\ &= \int \frac{1}{p(O)}\log q_\phi(\tau)\frac{p(\tau,O)}{q_\phi(\tau)}q_\phi(\tau)d\tau + c = \frac{1}{Z}\mathop{\mathbb{E}}_{\tau\sim q_\phi}\left[\log q_\phi(\tau)\frac{p(\tau,O)}{q_\phi(\tau)}\right] + c \end{aligned}$$

4.0.2 Un-marginalized gradient of the Forward KL Objective

Crucially, because the original problem is over a distribution not dependent on ϕ, we can directly pass the gradient into the expectation without the REINFORCE trick, or reparameterization:

$$\nabla_\phi KL(p(\tau|O=1)||q_\phi(\tau)) = \mathop{\mathbb{E}}_{\tau\sim q_\phi}\left[\frac{p(\tau,O)}{q_\phi(\tau)}\frac{\nabla_\phi \log q_\phi(\tau)}{Z}\right].$$

This actually gives us a simple algorithm that can evaluate the gradient, and thereby minimize the expected KL between our two distributions following samples generated from our policy interacting with the simulator. Based upon this approach, we can apply self normalized importance weighting to avoid explicitly computing the constant Z, at the cost of a biased but consistent estimator:

$$\nabla_\phi KL(p(\tau|O)||q_\phi(\tau)) \approx \sum_{i=1}^{m}\left[w_i\nabla_\phi \log q_\phi(\tau_i)\right] = \frac{1}{Z}\sum_{i=1}^{m}\left[\frac{p(\tau_i,O_i)}{q_\phi(\tau_i)}\nabla_\phi \log q_\phi(\tau_i)\right].$$

Where for optimal trajectories ($O = O_{t=0:T-1} = 1$), the un-normalized weights are defined as,

$$w_i = \prod_{t=0}^{T-1}\frac{\frac{1}{Z_0}\exp r_t(a_t,s_t)\pi_0(a_t|s_t)}{\pi_\phi(a_t|s_t)} = \frac{1}{\hat{Z}_0}\exp\left[\sum_{t=0}^{T-1} r(a_t^i,s_t^i) - \log\left(\frac{\pi_\phi(a_t^i|s_t^i)}{\pi_0(a_t^i|s_t^i)}\right)\right]$$

We also include reward normalization constant Z_0 for generality. In order to avoid computation of Z explicitly, we take advantage of the consistent but biased self normalized importance weighting estimator. This means weights w_i are replaced with $\hat{w_i}$ using $\tau_i \sim q_\phi(\tau)$:

$$\hat{w_i} = \frac{w_i}{\sum_{j=1}^m w_j} = \frac{\exp\left[\sum_{t=0}^{T-1} r(a_t^i,s_t^i) - \log\left(\frac{\pi_\phi(a_t^i|s_t^i)}{\pi_0(a_t^i|s_t^i)}\right)\right]}{\sum_{j=1}^m \exp\left[\sum_{t=0}^{T-1} r(a_t^j,s_t^j) - \log\left(\frac{\pi_\phi(a_t^j|s_t^j)}{\pi_0(a_t^j|s_t^j)}\right)\right]}. \tag{8}$$

In this case we now have a biased estimator of the gradient, but also one that is generally low variance when compared to its un-marginalized RKL counterpart [34],

$$\nabla_\phi KL(p(\tau|O)||q_\phi(\tau)) \approx \sum_{i=1}^{n}\frac{\exp\left[\sum_{t=0}^{T-1} r(a_t^i,s_t^i) - \log\left(\frac{\pi_\phi(a_t^i|s_t^i)}{\pi_0(a_t^i|s_t^i)}\right)\right]}{\sum_{j=1}^n \exp\left[\sum_{t=0}^{T-1} r(a_t^j,s_t^j) - \log\left(\frac{\pi_\phi(a_t^j|s_t^j)}{\pi_0(a_t^j|s_t^j)}\right)\right]}\nabla_\phi \log q_\phi(\tau_i) \tag{9}$$

4.0.3 Marginalization Proof for forward KL

Like the reverse KL, we can further reduce variance of our gradient estimator. In order to show that the objective can be marginalized with respect to actions given states and optimality, we start with the original definition of the objectives gradient. Using d separation of the probabilistic graphical model to ignore the optimality variables prior to the state conditioned on we have that:

$$\nabla_\phi J_{p,q}(\phi) = \sum_{t=0}^{T-1} \int_\tau p(a_0, s_0, ...s_T | o_t, ..., o_{T-1}, s_t) p(s_t | o_0, ..., o_{T-1}) \nabla_\phi \log \pi_\phi(a_t|s_t) d\tau.$$

Next looking at just one element of this sum at index t,

$$\begin{aligned}
&= \int_{s_t} ... \int_{s_T} \int_{a_t} ... \int_{a_{T-1}} p(s_t|O_0...O_{T-1}) \nabla_\phi \log \pi_\phi(a_t|s_t) \\
&\quad ... \int_{s_0} ... \int_{s_{t-1}} \int_{a_0} ... \int_{a_{t-1}} p(a_0, s_0, ...s_T | o_t, ..., o_T, s_t) d\tau \\
&= \int_{s_t} ... \int_{s_T} \int_{a_t} ... \int_{a_{T-1}} p(s_t|O_0...O_{T-1}) \nabla_\phi \log \pi_\phi(a_t|s_t) p(a_t, s_{t+1}...s_T | o_t, ..., o_T, s_t) d\tau \\
&= \int_{s_t} ... \int_{s_T} \int_{a_t} ... \int_{a_{T-1}} p(s_t|O_0...O_{T-1}) \nabla_\phi \log \pi_\phi(a_t|s_t) \frac{p(a_t, s_{t+1}...s_T, o_t, ..., o_T | s_t)}{p(o_t, ..., o_T|s_t)} d\tau \\
&= \int_{s_t} ... \int_{s_T} \int_{a_t} ... \int_{a_{T-1}} \frac{p(s_t|O_0...O_{T-1}) \pi_0(a_t|s_t)}{p(o_t, ..., o_T|s_t)} \nabla_\phi \log \pi_\phi(a_t|s_t) p(s_{t+1}...s_T, o_t, ..., o_T | s_t, a_t) d\tau \\
&= \int_{s_t} ... \int_{s_T} \int_{a_t} ... \int_{a_{T-1}} \frac{p(s_t|O_0...O_{T-1}) \pi_0(a_t|s_t)}{p(o_t, ..., o_T|s_t)} \nabla_\phi \log \pi_\phi(a_t|s_t) p(s_{t+1}...s_T, o_t, ..., o_T | s_t, a_t) d\tau \\
&= \mathop{\mathbb{E}}_{p(s_t|O_{0:T})} \left[\frac{1}{p(O_{t:T}|s_t)} \mathop{\mathbb{E}}_{a_t \sim \pi_0(a_t|s_t)} \left[\nabla_\phi \log \pi_\phi(a_t|s_t) \mathop{\mathbb{E}}_{q_\phi(\tau_{t+1:T}|a_t, s_t)} \left[\frac{p(\tau_{t+1:T}, O_{t:T}|a_t, s_t)}{q_\phi(\tau_{t+1:T}|a_t, s_t)} \right] \right] \right]
\end{aligned}$$

Notably, we can define a specific expectation for convenience, which can be either be estimated through sampling or a learned function approximator as is done in practice [2, 47]:

$$\begin{aligned}
&\mathop{\mathbb{E}}_{q_\phi(\tau_{t+1:T}|s_t, a_t)} \left[\frac{p(\tau_{t+1:T}, O_{t:T}|a_t, s_t)}{q(\tau_{t+1:T}|a_t, s_t)} \right] = \mathop{\mathbb{E}}_{q_\phi(\tau_{t+1:T}|s_t, a_t)} \left[\frac{p(\tau_{t+1:T}, O_{t:T}|a_t, s_t)}{q_\phi(\tau_{t+1:T}|a_t, s_t)} \right] \\
&= \mathop{\mathbb{E}}_{q_\phi(\tau_{t+1:T}|s_t, a_t)} \left[\exp \left[r(a_t, s_t) + \sum_{t'=t+1}^{T} r(a_{t'}, s_{t'}) - \frac{\pi_\phi(a_{t'}|s_{t'})}{\pi_0(a_{t'}|s_{t'})} \right] \right] = \exp(\hat{Q}(a_t, s_t))
\end{aligned}$$

We then note the following about the normalization constant sitting outside the action expectation:

$$p(O_{t:T}|s_t) = \int p(O_{t:T}, a_t|s_t) da_t = \int p(O_{t:T}|a_t, s_t) \pi_0(a_t|s_t) da_t = \mathop{\mathbb{E}}_{a_t \sim \pi_0(a_t|s_t)} \left[\exp(\hat{Q}(a_t, s_t)) \right]$$

This definition allows us to define our marginalized expectation more simply, and identify the crucial issue with direct estimation of the marginalized forward KL:

$$\nabla_\phi J_{p,q}(\phi) = \mathop{\mathbb{E}}_{p(s_t|O_{0:T})} \left[\mathop{\mathbb{E}}_{\pi_\phi(a_t|s_t)} \left[\frac{\nabla_\phi \log \pi_\phi(a_t|s_t) \exp \left[\hat{Q}(a_t, s_t) - \log \left[\frac{\pi_\phi(a_t|s_t)}{\pi_0(a_t|s_t)} \right] \right]}{\mathbb{E}_{a'_t \sim \pi_\phi(a'_t|s_t)} [\exp \hat{Q}(a'_t, s_t) - \log \left[\frac{\pi_\phi(a_t|s_t)}{\pi_0(a_t|s_t)} \right]]} \right] \right]$$

We can deal with the outer expectation in terms of $p(s_t|O_{0:T})$ as well, though it induces a less satisfactory estimator whose variance again grows with time-horizon:

$$\begin{aligned}
&\mathop{\mathbb{E}}_{p(s_t|O_{0:T})} [f(s_t)] = \int f(s_t) p(s_t|O_{0:T}) ds_t = \int f(s_t) \frac{p(s_t, O_{0:T})}{p(O_{0:T})} ds_t = \int f(s_t) p(s_t) \frac{p(O_{0:T}|s_t)}{p(O_{0:T})} ds_t \\
&= \frac{1}{p(O_{0:T})} \mathop{\mathbb{E}}_{p(s_t)} [p(O_{0:T}|s_t) f(s_t)] = \frac{\int p(O_{0:T}, s_t) f(s_t) ds_t}{\int p(O_{0:T}, s_t) ds_t} = \frac{\int p(O_{0:T}, \tau_{0:T}) f(s_t) d\tau_{0:T}}{\int p(O_{0:T}, \tau_{0:T}) d\tau_{0:T}} \\
&= \mathop{\mathbb{E}}_{q_\phi(\tau_{0:T})} \left[\frac{p(O_{0:T}, \tau_{0:T})}{q_\phi(\tau_{0:T})} f(s_t) \right] \Big/ \mathop{\mathbb{E}}_{q_\phi(\tau_{0:T})} \left[\frac{p(O_{0:T}, \tau_{0:T})}{q_\phi(\tau_{0:T})} \right]
\end{aligned}$$

Such an estimator would function by sampling N independent trajectories from q_ϕ, and then for the specified time, re-weight each function f at its respective state by its associated regularized cumulative return. With this consideration in mind, we can define importance weights over states and actions for each time-step which produce a consistent estimate of the forward KL gradient:

$$\nabla_\phi J_{p,q}(\phi) \approx \sum_{t=0}^{T} \sum_{i=1}^{M} \bar{y}_i^t \sum_{k=1}^{N} \bar{x}_{i,k}^t \nabla_\phi \log \pi_\phi(a_t^k|s_t^i), \quad \text{with } \tau^i \sim q_\phi(\tau),\ a_t^k \sim \pi_\phi(a_t|s_t^i)$$

$$\text{where} \quad \bar{y}_i^t = \frac{y_i^t}{\sum_{j=1}^{M} y_j^t} \quad \text{with} \quad y_i^t = \exp\left[\sum_{t'=0}^{T-1} r(a_{t'}^i, s_{t'}^i) - \log\left(\frac{\pi_\phi(a_{t'}^i|s_{t'}^i)}{\pi_0(a_{t'}^i|s_{t'}^i)}\right)\right]$$

$$\text{and} \quad \bar{x}_{i,k}^t = \frac{x_{i,k}^t}{\sum_{j=1}^{N} x_{i,j}^t} \quad \text{with} \quad x_{i,k}^t = \exp\left[\hat{Q}(a_t^k, s_t^i) - \log\left[\frac{\pi_\phi(a_t^k|s_t^i)}{\pi_0(a_t^k|s_t^i)}\right]\right]$$

For our experiments, as done in practice in [2, 47], we re-weight the loss each at each state uniformly (i.e. $\bar{y}_i^t = \frac{1}{M}$), to avoid high variance of the gradient estimator, ensure proper coverage of the state-space for better generalization, and to maintain sample efficiency within the larger RL algorithm. Additionally we target the classic Q function instead of the soft Q function $\bar{Q}$, as we found that using this lower bound on $\bar{Q}$ tended to produce more stable learning within the policy as in prior work [2].

5 Experiments

To display the relative merits of both objectives, we include experiments where a native implementation of soft actor critic was taken, and modified to include the forward KL objective (FKL). We begin with an illustration of an instance where you can find improvement by replacing the more common RKL for the FKL objective. In Figure 2, we plot results of training on the Humanoid benchmark from MuJoCo using RKL and FKL. Here, the FKL improves sample efficiency over the RKL, demonstrating approximately 60% increase in sample efficiency. Humanoid is the most challenging of the five tasks considered from the MuJoCo suite.

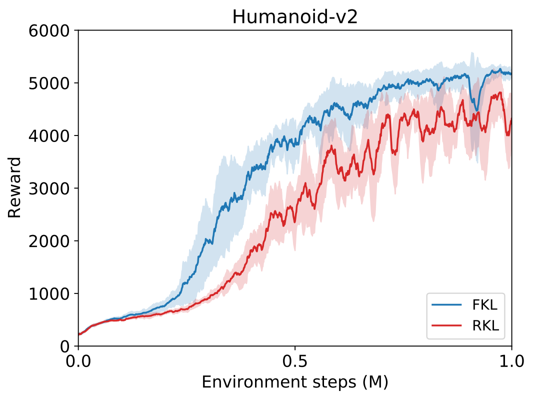

Figure 2: Comparison of RKL and FKL on the MuJoCo Humanoid benchmark. Error bars denote one standard deviation. As shown, the reward garnered throughout optimization is also lower variance trained with FKL than RKL.

Additional advantages of the FKL policy objective include its ease of implementation and its distribution agnostic property. This property means that in discrete control, or for hierarchical policies with discrete branching, the RKL objective would require further modification, while the FKL functions as is. One drawback that we note, however, is that the FKL estimator scales with computation, i.e. its bias and thus its performance is dependent on the number of N re-samples from the policy. Finally, for continuous control, we additionally show that the FKL can also improve the algorithm's robustness to changes in hyper-parameters. All experiments are run with at 10 replicates and plotted with one-half standard deviation and smoothed using a 0.9 weighted EMA. As seen in Figure 3, the forward KL can maintain policy performance, even when memory is severely limited.

6 Related Work

There is a growing literature on various methods that incorporate the RLAI framework discussed within this paper [2, 24, 35, 59, 60, 31, 43, 55]. In many cases these works use RLAI to give theoretical basis for the empirical effectiveness of well established implementation tricks such as entropy bonuses [24, 35, 59]. Very few works however, have considered the comparison between the forward and reverse KL as well as the relative merits of each in the context of reinforcement learning objectives [43, 38]. Even fewer produce scalable and sample efficient implementations of both objectives, making a proper comparison difficult. We note for completeness that work has also

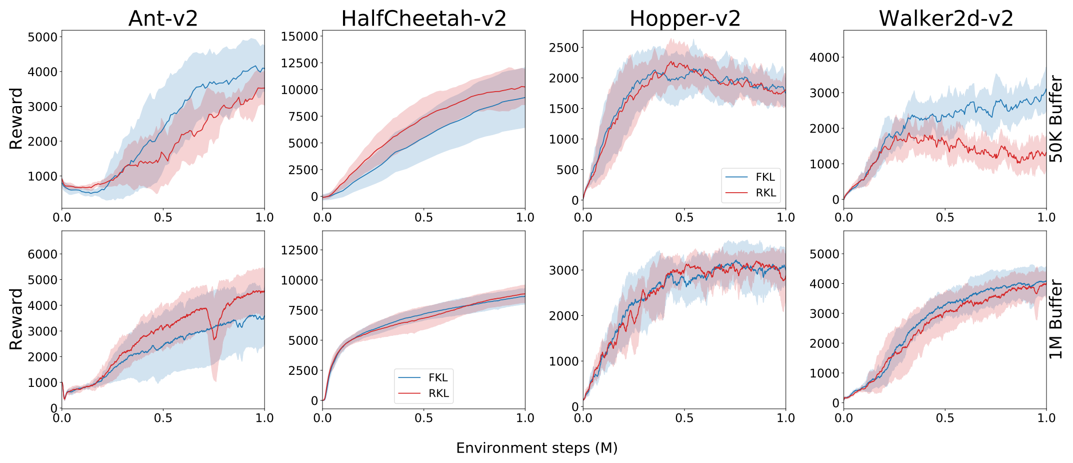

Figure 3: First row: comparison of RKL and FKL on MuJoCo benchmark tasks using a base implementation of SAC *without* hyper-parameter tuning for MuJoCo. The notable difference in hyper-parameters between these settings is the buffer sizes of 50K versus 1M for the first and second rows, respectively. We find under these conditions, FKL is always competitive or significantly better than the baseline with the exception of the Ant benchmark using 1M replay buffer size. In all figures, error bars denote one-quarter standard deviation.

been done in formalizing methods relating to the original policy gradient algorithm [58, 50, 48] using tools from the optimization literature [21, 56], though these analysis do not directly relate the the RLAI framework. Work by [7] more closely relates to this work in their analysis of improvement under the FKL, however they do not focus on practical gradient estimation. Similarly there has been work analyzing critic and gradient estimation [9] in average return MDPs, as well as empirical studies for the more commonly utilized discounted MDPs discussed above [29].

Algorithmically, the closest class of algorithms to our work in the classical literature comes from policy search methods [43, 42, 36, 11, 46, 12]. Of the subset of these that have been made scalable, closely related algorithms to our work come from [2, 51, 8]. Unlike [2], we do not require an alternating update and directly learn the policy via the forward KL. Additionally while we do not provide monotonic improvement guarantees which require additional assumptions and a trust-region update, we do provide a full proof of marginalization of the forward KL as it applies to RLAI in a finite horizon setting. A similar proof for a marginalization can be found in [47], however this assumes access to the steady state distribution of the Markov chain as well as a model of the environment. We do note that the objective for the target policy given in [51] does come very close to our own, however it requires additional parameterization of a value function, as well as a more complicated alternating update for the temperature coefficient α, and a different inference formulation which culminates in an advantage based objective. Most importantly, while these works represent end-to-end algorithms, our estimators can be applied a wide variety of existing frameworks.

7 Discussion

This paper describes in detail the differences in policy objective which occur both mathematically, and practically under a divergence minimization perspective of RLAI. Crucially, we show that there are very real advantages to considering other objectives besides the classic policy gradient methods which are most common. While these alternatives like the FKL are generally not going to drastically change performance, their ease of implementation, robustness to hyperparameters, and their distribution agnostic nature make them a very desirable alternative to classic methods. In the future we would like to test how these objectives function on a wider variety of benchmarks and policy distributions. Additionally, its possible that better non-uniform sampling methods might be derived following $p(s_t|O_{0:T})$. Though estimating an expectation under such a distribution might be difficult, and potentially require investigation into the particle smoothing literature [13].

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. *arXiv preprint arXiv:1806.06920*, 2018.

[3] Samuel Ainsworth, Kendall Lowrey, John Thickstun, Zaid Harchaoui, and Siddhartha Srinivasa. Faster policy learning with continuous-time gradients, 2021.

[4] Dimitri P Bertsekas. *Reinforcement learning and optimal control*. Athena Scientific, 2019.

[5] Jeff Bezanson, Stefan Karpinski, Viral B Shah, and Alan Edelman. Julia: A fast dynamic language for technical computing. *arXiv preprint arXiv:1209.5145*, 2012.

[6] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

[7] Alan Chan, Hugo Silva, Sungsu Lim, Tadashi Kozuno, A. Rupam Mahmood, and Martha White. Greedification operators for policy optimization: Investigating forward and reverse kl divergences, 2021.

[8] Yinlam Chow, Brandon Cui, MoonKyung Ryu, and Mohammad Ghavamzadeh. Variational model-based policy optimization. *arXiv preprint arXiv:2006.05443*.

[9] Wesley Chung, Valentin Thomas, Marlos C. Machado, and Nicolas Le Roux. Beyond variance reduction: Understanding the true impact of baselines on policy optimization. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 1999–2009. PMLR, 18–24 Jul 2021.

[10] Ignasi Clavera, Jonas Rothfuss, John Schulman, Yasuhiro Fujita, Tamim Asfour, and Pieter Abbeel. Model-based reinforcement learning via meta-policy optimization, 2018.

[11] Adriá Colomé and Carme Torras. Dual reps: A generalization of relative entropy policy search exploiting bad experiences. *IEEE Transactions on Robotics*, 33(4):978–985, 2017.

[12] Marc Peter Deisenroth, Gerhard Neumann, and Jan Peters. *A survey on policy search for robotics*. now publishers, 2013.

[13] Arnaud Doucet, Adam M Johansen, et al. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, 12(656-704):3, 2009.

[14] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control, 2016.

[15] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep rl: A case study on ppo and trpo. In *International Conference on Learning Representations*, 2019.

[16] Matthew Fellows, Anuj Mahajan, Tim G. J. Rudner, and Shimon Whiteson. Virel: A variational inference framework for reinforcement learning, 2018.

[17] Matthew Fellows, Anuj Mahajan, Tim GJ Rudner, and Shimon Whiteson. Virel: A variational inference framework for reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 7120–7134, 2019.

[18] C. Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. Brax - a differentiable physics engine for large scale rigid body simulation. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[19] Scott Fujimoto, Herke Van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*, 2018.

[20] Alexandre Galashov, Siddhant M Jayakumar, Leonard Hasenclever, Dhruva Tirumala, Jonathan Schwarz, Guillaume Desjardins, Wojciech M Czarnecki, Yee Whye Teh, Razvan Pascanu, and Nicolas Heess. Information asymmetry in kl-regularized rl. *arXiv preprint arXiv:1905.01240*, 2019.

[21] Dibya Ghosh, Marlos C. Machado, and Nicolas Le Roux. An operator view of policy gradient methods, 2020.

[22] Shixiang Gu, Timothy Lillicrap, Zoubin Ghahramani, Richard E Turner, and Sergey Levine. Q-prop: Sample-efficient policy gradient with an off-policy critic. *arXiv preprint arXiv:1611.02247*, 2016.

[23] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies, 2017.

[24] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

[25] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications, 2018.

[26] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters, 2017.

[27] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[28] Maximilian Igl, Luisa Zintgraf, Tuan Anh Le, Frank Wood, and Shimon Whiteson. Deep variational reinforcement learning for pomdps. *arXiv preprint arXiv:1806.02426*, 2018.

[29] Andrew Ilyas, Logan Engstrom, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. A closer look at deep policy gradients. *arXiv preprint arXiv:1811.02553*, 2018.

[30] Riashat Islam, Peter Henderson, Maziar Gomrokchi, and Doina Precup. Reproducibility of benchmarked deep reinforcement learning tasks for continuous control, 2017.

[31] Hilbert J Kappen, Vicenç Gómez, and Manfred Opper. Optimal control as a graphical model inference problem. *Machine learning*, 87(2):159–182, 2012.

[32] Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-ensemble trust-region policy optimization. *arXiv preprint arXiv:1802.10592*, 2018.

[33] Tuan Anh Le, Maximilian Igl, Tom Rainforth, Tom Jin, and Frank Wood. Auto-encoding sequential monte carlo. *arXiv preprint arXiv:1705.10306*, 2017.

[34] Tuan Anh Le, Adam R. Kosiorek, N. Siddharth, Yee Whye Teh, and Frank Wood. Revisiting reweighted wake-sleep for models with stochastic control flow, 2018.

[35] Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

[36] Sergey Levine and Vladlen Koltun. Guided policy search. In *International Conference on Machine Learning*, pages 1–9, 2013.

[37] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[38] Xubo Lyu and Mo Chen. Ttr-based rewards for reinforcement learning with implicit model priors. *arXiv preprint arXiv:1903.09762*, 2019.

[39] Luke Metz, C. Daniel Freeman, Samuel S. Schoenholz, and Tal Kachman. Gradients are not all you need, 2021.

[40] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

[41] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[42] William H Montgomery and Sergey Levine. Guided policy search via approximate mirror descent. In *Advances in Neural Information Processing Systems*, pages 4008–4016, 2016.

[43] Gerhard Neumann et al. Variational inference for policy search in changing situations. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, pages 817–824, 2011.

[44] Brendan O'Donoghue, Ian Osband, and Catalin Ionescu. Making sense of reinforcement learning and probabilistic inference. *arXiv preprint arXiv:2001.00805*, 2020.

[45] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[46] Jan Peters, Katharina Mulling, and Yasemin Altun. Relative entropy policy search. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[47] Alexandre Piché, Valentin Thomas, Cyril Ibrahim, Yoshua Bengio, and Chris Pal. Probabilistic planning with sequential monte carlo methods. In *International Conference on Learning Representations*, 2019.

[48] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

[49] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[50] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[51] H Francis Song, Abbas Abdolmaleki, Jost Tobias Springenberg, Aidan Clark, Hubert Soyer, Jack W Rae, Seb Noury, Arun Ahuja, Siqi Liu, Dhruva Tirumala, et al. V-mpo: On-policy maximum a posteriori policy optimization for discrete and continuous control. *arXiv preprint arXiv:1909.12238*, 2019.

[52] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press, 2011.

[53] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018.

[54] George Tucker, Surya Bhupatiraju, Shixiang Gu, Richard E Turner, Zoubin Ghahramani, and Sergey Levine. The mirage of action-dependent baselines in reinforcement learning. *arXiv preprint arXiv:1802.10031*, 2018.

[55] Jan-Willem Van De Meent, Brooks Paige, David Tolpin, and Frank Wood. Black-box policy search with probabilistic programs. 2016.

[56] Sharan Vaswani, Olivier Bachem, Simone Totaro, Robert Mueller, Matthieu Geist, Marlos C Machado, Pablo Samuel Castro, and Nicolas Le Roux. A functional mirror ascent view of policy gradient methods with function approximation. *arXiv preprint arXiv:2108.05828*, 2021.

[57] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pages 1995–2003, 2016.

[58] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, May 1992.

[59] Brian D. Ziebart. *Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy*. PhD thesis, Carnegie Mellon University, USA, 2010.

[60] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.