# Reinforcement Learning and Control as Probabilistic Inference: Tutorial and Review

**Sergey Levine**
UC Berkeley
svlevine@eecs.berkeley.edu

## Abstract

The framework of reinforcement learning or optimal control provides a mathematical formalization of intelligent decision making that is powerful and broadly applicable. While the general form of the reinforcement learning problem enables effective reasoning about uncertainty, the connection between reinforcement learning and inference in probabilistic models is not immediately obvious. However, such a connection has considerable value when it comes to algorithm design: formalizing a problem as probabilistic inference in principle allows us to bring to bear a wide array of approximate inference tools, extend the model in flexible and powerful ways, and reason about compositionality and partial observability. In this article, we will discuss how a generalization of the reinforcement learning or optimal control problem, which is sometimes termed maximum entropy reinforcement learning, is equivalent to exact probabilistic inference in the case of deterministic dynamics, and variational inference in the case of stochastic dynamics. We will present a detailed derivation of this framework, overview prior work that has drawn on this and related ideas to propose new reinforcement learning and control algorithms, and describe perspectives on future research.

## 1  Introduction

Probabilistic graphical models (PGMs) offer a broadly applicable and useful toolbox for the machine learning researcher (Koller and Friedman, 2009): by couching the entirety of the learning problem in the parlance of probability theory, they provide a consistent and flexible framework to devise principled objectives, set up models that reflect the causal structure in the world, and allow a common set of inference methods to be deployed against a broad range of problem domains. Indeed, if a particular learning problem can be set up as a probabilistic graphical model, this can often serve as the first and most important step to solving it. Crucially, in the framework of PGMs, it is sufficient to write down the model and pose the question, and the objectives for learning and inference emerge automatically.

Conventionally, decision making problems formalized as reinforcement learning or optimal control have been cast into a framework that aims to generalize probabilistic models by augmenting them with utilities or rewards, where the reward function is viewed as an extrinsic signal. In this view, determining an optimal course of action (a plan) or an optimal decision-making strategy (a policy) is a fundamentally distinct type of problem than probabilistic inference, although the underlying dynamical system might still be described by a probabilistic graphical model. In this article, we instead derive an alterate view of decision making, reinforcement learning, and optimal control, where the decision making problem is simply an inference problem in a particular type of graphical model. Formalizing decision making as inference in probabilistic graphical models can in principle allow us to to bring to bear a wide array of approximate inference tools, extend the model in flexible and powerful ways, and reason about compositionality and partial observability.

Specifically, we will discuss how a generalization of the reinforcement learning or optimal control problem, which is sometimes termed maximum entropy reinforcement learning, is equivalent to exact probabilistic inference in the case of deterministic dynamics, and variational inference in the case of stochastic dynamics. This observation is not a new one, and the connection between probabilistic inference and control has been explored in the literature under a variety of names, including

the Kalman duality (Todorov, 2008), maximum entropy reinforcement learning (Ziebart, 2010), KL-divergence control (Kappen et al., 2012; Kappen, 2011), and stochastic optimal control (Toussaint, 2009). While the specific derivations the differ, the basic underlying framework and optimization objective are the same. All of these methods involve formulating control or reinforcement learning as a PGM, either explicitly or implicitly, and then deploying learning and inference methods from the PGM literature to solve the resulting inference and learning problems.

Formulating reinforcement learning and decision making as inference provides a number of other appealing tools: a natural exploration strategy based on entropy maximization, effective tools for inverse reinforcement learning, and the ability to deploy powerful approximate inference algorithms to solve reinforcement learning problems. Furthermore, the connection between probabilistic inference and control provides an appealing probabilistic interpretation for the meaning of the reward function, and its effect on the optimal policy. The design of reward or cost functions in reinforcement learning is oftentimes as much art as science, and the choice of reward often blurs the line between algorithm and objective, with task-specific heuristics and task objectives combined into a single reward. In the control as inference framework, the reward induces a distribution over random variables, and the optimal policy aims to explicitly match a probability distribution defined by the reward and system dynamics, which may in future work suggest a way to systematize reward design.

This article will present the probabilistic model that can be used to embed a maximum entropy generalization of control or reinforcement learning into the framework of PGMs, describe how to perform inference in this model – exactly in the case of deterministic dynamics, or via structured variational inference in the case of stochastic dynamics, – and discuss how approximate methods based on function approximation fit within this framework. Although the particular variational inference interpretation of control differs somewhat from the presentation in prior work, the goal of this article is not to propose a fundamentally novel way of viewing the connection between control and inference. Rather, it is to provide a unified treatment of the topic in a self-contained and accessible tutorial format, and to connect this framework to recent research in reinforcement learning, including recently proposed deep reinforcement learning algorithms. In addition, this article presents a review of the recent reinforcement learning literature that relates to this view of control as probabilistic inference, and offers some perspectives on future research directions.

The basic graphical model for control will be presented in Section 2, variational inference for stochastic dynamics will be discussed in Section 3, approximate methods based on function approximation, including deep reinforcement learning, will be discussed in Section 4, and a survey and review of recent literature will be presented in Section 5. Finally, we will discuss perspectives on future research directions in Section 6.

## 2 A Graphical Model for Control as Inference

In this section, we will present the basic graphical model that allows us to embed control into the framework of PGMs, and discuss how this framework can be used to derive variants of several standard reinforcement learning and dynamic programming approaches. The PGM presented in this section corresponds to a generalization of the standard reinforcement learning problem, where the RL objective is augmented with an entropy term. The magnitude of the reward function trades off between reward maximization and entropy maximization, allowing the original RL problem to be recovered in the limit of infinitely large rewards. We will begin by defining notation, then defining the graphical model, and then presenting several inference methods and describing how they relate to standard algorithms in reinforcement learning and dynamic programming. Finally, we will discuss a few limitations of this method and motivate the variational approach in Section 3.

### 2.1 The Decision Making Problem and Terminology

First, we will introduce the notation we will use for the standard optimal control or reinforcement learning formulation. We will use $\mathbf{s} \in \mathcal{S}$ to denote states and $\mathbf{a} \in \mathcal{A}$ to denote actions, which may each be discrete or continuous. States evolve according to the stochastic dynamics $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, which are in general unknown. We will follow a discrete-time finite-horizon derivation, with horizon $T$, and omit discount factors for now. A discount $\gamma$ can be readily incorporated into this framework simply by modifying the transition dynamics, such that any action produces a transition into an absorbing state with probability $1 - \gamma$, and all standard transition probabilities are multiplied by $\gamma$.

(a) graphical model with states and actions
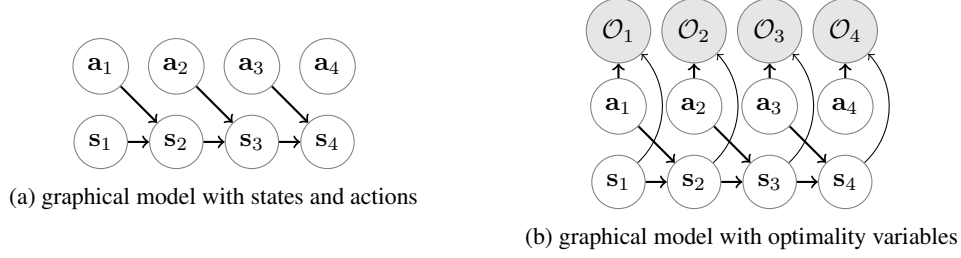


(b) graphical model with optimality variables

Figure 1: The graphical model for control as inference. We begin by laying out the states and actions, which form the backbone of the model (a). In order to embed a control problem into this model, we need to add nodes that depend on the reward (b). These "optimality variables" correspond to observations in a HMM-style framework: we condition on the optimality variables being true, and then infer the most probable action sequence or most probable action distributions.

A task in this framework can be defined by a reward function $r(\mathbf{s}_t, \mathbf{a}_t)$. Solving a task typically involves recovering a policy $p(\mathbf{a}_t|\mathbf{s}_t, \theta)$, which specifies a distribution over actions conditioned on the state parameterized by some parameter vector $\theta$. A standard reinforcement learning policy search problem is then given by the following maximization:

$$\theta^\star = \arg\max_\theta \sum_{t=1}^{T} E_{(\mathbf{s}_t, \mathbf{a}_t) \sim p(\mathbf{s}_t, \mathbf{a}_t | \theta)}[r(\mathbf{s}_t, \mathbf{a}_t)]. \tag{1}$$

This optimization problem aims to find a vector of policy parameters $\theta$ that maximize the total expected reward $\sum_t r(\mathbf{s}_t, \mathbf{a}_t)$ of the policy. The expectation is taken under the policy's *trajectory* distribution $p(\tau)$, given by

$$p(\tau) = p(\mathbf{s}_1, \mathbf{a}_t, \dots, \mathbf{s}_T, \mathbf{a}_T | \theta) = p(\mathbf{s}_1) \prod_{t=1}^{T} p(\mathbf{a}_t | \mathbf{s}_t, \theta) p(\mathbf{s}_{t+1} | \mathbf{s}_t, \mathbf{a}_t). \tag{2}$$

For conciseness, it is common to denote the action conditional $p(\mathbf{a}_t|\mathbf{s}_t, \theta)$ as $\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$, to emphasize that it is given by a parameterized policy with parameters $\theta$. These parameters might correspond, for example, to the weights in a neural network. However, we could just as well embed a standard planning problem in this formulation, by letting $\theta$ denote a sequence of actions in an open-loop plan.

Having formulated the decision making problem in this way, the next question we have to ask to derive the control as inference framework is: how can we formulate a probabilistic graphical model such that the most probable trajectory corresponds to the trajectory from the optimal policy? Or, equivalently, how can we formulate a probabilistic graphical model such that inferring the posterior action conditional $p(\mathbf{a}_t|\mathbf{s}_t, \theta)$ gives us the optimal policy?

## 2.2 The Graphical Model

To embed the control problem into a graphical model, we can begin simply by modeling the relationship between states, actions, and next states. This relationship is simple, and corresponds to a graphical model with factors of the form $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, as shown in Figure 1 (a). However, this graphical model is insufficient for solving control problems, because it has no notion of rewards or costs. We therefore have to introduce an additional variable into this model, which we will denote $\mathcal{O}_t$. This additional variable is a binary random variable, where $\mathcal{O}_t = 1$ denotes that time step $t$ is *optimal*, and $\mathcal{O}_t = 0$ denotes that it is not optimal. We will choose the distribution over this variable to be given by the following equation:

$$p(\mathcal{O}_t = 1|\mathbf{s}_t, \mathbf{a}_t) = \exp(r(\mathbf{s}_t, \mathbf{a}_t)). \tag{3}$$

The graphical model with these additional variables is summarized in Figure 1 (b). While this might at first seem like a peculiar and arbitrary choice, it leads to a very natural posterior distribution over

3

actions when we condition on $\mathcal{O}_t = 1$ for all $t \in \{1, \ldots, T\}$:

$$p(\tau|\mathbf{o}_{1:T}) \propto p(\tau, \mathbf{o}_{1:T}) = p(\mathbf{s}_1) \prod_{t=1}^{T} p(\mathcal{O}_t = 1|\mathbf{s}_t, \mathbf{a}_t) p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$$

$$= p(\mathbf{s}_1) \prod_{t=1}^{T} \exp(r(\mathbf{s}_t, \mathbf{a}_t)) p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$$

$$= \left[ p(\mathbf{s}_1) \prod_{t=1}^{T} p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) \right] \exp\left( \sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t) \right). \tag{4}$$

That is, the probability of observing a given trajectory is given by the product between its probability to occur according to the dynamics (the term in square brackets on the last line), and the exponential of the total reward along that trajectory. It is most straightforward to understand this equation in systems with deterministic dynamics, where the first term is a constant for all trajectories that are dynamically feasible. In this case, the trajectory with the highest reward has the highest probability, and trajectories with lower reward have exponentially lower probability. If we would like to plan for an optimal action sequence starting from some initial state $\mathbf{s}_1$, we can condition on $\mathbf{o}_{1:T}$ and choose $p(\mathbf{s}_1) = \delta(\mathbf{s}_1)$, in which case maximum a posteriori inference corresponds to a kind of planning problem. It is easy to see that this exactly corresponds to standard planning or trajectory optimization in the case where the dynamics are deterministic, in which case Equation (4) reduces to

$$p(\tau|\mathbf{o}_{1:T}) \propto \mathbb{1}[p(\tau) \neq 0] \exp\left( \sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t) \right). \tag{5}$$

Here, the indicator function simply indicates that the trajectory $\tau$ is dynamically consistent (meaning that $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) \neq 0$) and the initial state is correct. The case of stochastic dynamics poses some challenges, and will be discussed in detail in Section 3. However, even under deterministic dynamics, we are often interested in recovering a policy rather than a plan. In this PGM, the optimal policy can be written as $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{t:T} = 1)$ (we will drop $= 1$ in the remainder of the derivation for conciseness). This distribution is somewhat analogous to $p(\mathbf{a}_t|\mathbf{s}_t, \theta^\star)$ in the previous section, with two major differences: first, it is independent of the parameterization $\theta$, and second, we will see later that it optimizes an objective that is slightly different from the standard reinforcement learning objective in Equation (1).

## 2.3  Policy Search as Probabilistic Inference

We can recover the optimal policy $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{t:T})$ using a standard sum-product inference algorithm, analogously to inference in HMM-style dynamic Bayesian networks. As we will see in this section, it is sufficient to compute backward messages of the form

$$\beta_t(\mathbf{s}_t, \mathbf{a}_t) = p(\mathcal{O}_{t:T}|\mathbf{s}_t, \mathbf{a}_t).$$

These messages have a natural interpretation: they denote the probability that a trajectory can be optimal for time steps from $t$ to $T$ if it begins in state $\mathbf{s}_t$ with the action $\mathbf{a}_t$.[1] Slightly overloading the notation, we will also introduce the message

$$\beta_t(\mathbf{s}_t) = p(\mathcal{O}_{t:T}|\mathbf{s}_t).$$

These messages denote the probability that the trajectory from $t$ to $T$ is optimal if it begins in state $\mathbf{s}_t$. We can recover the state-only message from the state-action message by integrating out the action:

$$\beta_t(\mathbf{s}_t) = p(\mathcal{O}_{t:T}|\mathbf{s}_t) = \int_{\mathcal{A}} p(\mathcal{O}_{t:T}|\mathbf{s}_t, \mathbf{a}_t) p(\mathbf{a}_t|\mathbf{s}_t) d\mathbf{a}_t = \int_{\mathcal{A}} \beta_t(\mathbf{s}_t, \mathbf{a}_t) p(\mathbf{a}_t|\mathbf{s}_t) d\mathbf{a}_t.$$

The factor $p(\mathbf{a}_t|\mathbf{s}_t)$ is the action *prior*. Note that it is not conditioned on $\mathcal{O}_{1:T}$ in any way: it does not denote the probability of an optimal action, but simply the prior probability of actions. The PGM in Figure 1 doesn't actually contain this factor, and we can assume that $p(\mathbf{a}_t|\mathbf{s}_t) = \frac{1}{|\mathcal{A}|}$ for simplicity – that is, it is a constant corresponding to a uniform distribution over the set of actions.

---

[1] Note that $\beta_t(\mathbf{s}_t, \mathbf{a}_t)$ is *not* a probability density over $\mathbf{s}_t, \mathbf{a}_t$, but rather the probability of $\mathcal{O}_{t:T} = 1$.

The recursive message passing algorithm for computing $\beta_t(\mathbf{s}_t, \mathbf{a}_t)$ proceeds from the last time step $t = T$ backward through time to $t = 1$. In the base case, we note that $p(\mathcal{O}_T|\mathbf{s}_T, \mathbf{a}_T)$ is simply proportional to $\exp(r(\mathbf{s}_T, \mathbf{a}_T))$, since there is only one factor to consider. The recursive case is then given as following:

$$\beta_t(\mathbf{s}_t, \mathbf{a}_t) = p(\mathcal{O}_{t:T}|\mathbf{s}_t, \mathbf{a}_t) = \int_{\mathcal{S}} \beta_{t+1}(\mathbf{s}_{t+1})p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t)d\mathbf{s}_{t+1}. \tag{6}$$

From these backward messages, we can then derive the optimal policy $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T})$. First, note that $\mathcal{O}_{1:(t-1)}$ is conditionally independent of $\mathbf{a}_t$ given $\mathbf{s}_t$, which means that $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T}) = p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{t:T})$, and we can disregard the past when considering the current action distribution. This makes intuitive sense: in a Markovian system, the optimal action does not depend on the past. From this, we can easily recover the optimal action distribution using the two backward messages:

$$p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{t:T}) = \frac{p(\mathbf{s}_t, \mathbf{a}_t|\mathcal{O}_{t:T})}{p(\mathbf{s}_t|\mathcal{O}_{t:T})} = \frac{p(\mathcal{O}_{t:T}|\mathbf{s}_t, \mathbf{a}_t)p(\mathbf{a}_t|\mathbf{s}_t)p(\mathbf{s}_t)}{p(\mathcal{O}_{t:T}|\mathbf{s}_t)p(\mathbf{s}_t)} \propto \frac{p(\mathcal{O}_{t:T}|\mathbf{s}_t, \mathbf{a}_t)}{p(\mathcal{O}_{t:T}|\mathbf{s}_t)} = \frac{\beta_t(\mathbf{s}_t, \mathbf{a}_t)}{\beta_t(\mathbf{s}_t)},$$

where the order of conditioning in the third step is flipped by using Bayes' rule, and cancelling the factor of $p(\mathcal{O}_{t:T})$ that appears in both the numerator and denominator. The term $p(\mathbf{a}_t|\mathbf{s}_t)$ disappears, since we previously assumed it was a uniform distribution.

This derivation provides us with a solution, but perhaps not as much of the intuition. The intuition can be recovered by considering what these equations are doing in log space. To that end, we will introduce the log-space messages as

$$Q(\mathbf{s}_t, \mathbf{a}_t) = \log \beta_t(\mathbf{s}_t, \mathbf{a}_t)$$
$$V(\mathbf{s}_t) = \log \beta_t(\mathbf{s}_t).$$

The use of $Q$ and $V$ here is not accidental: the log-space messages correspond to "soft" variants of the state and state-action value functions. First, consider the marginalization over actions in log-space:

$$V(\mathbf{s}_t) = \log \int_{\mathcal{A}} \exp(Q(\mathbf{s}_t, \mathbf{a}_t))d\mathbf{a}_t.$$

When the values of $Q(\mathbf{s}_t, \mathbf{a}_t)$ are large, the above equation resembles a hard maximum over $\mathbf{a}_t$. That is, for large $Q(\mathbf{s}_t, \mathbf{a}_t)$,

$$V(\mathbf{s}_t) = \log \int_{\mathcal{A}} \exp(Q(\mathbf{s}_t, \mathbf{a}_t))d\mathbf{a}_t \approx \max_{\mathbf{a}_t} Q(\mathbf{s}_t, \mathbf{a}_t).$$

For smaller values of $Q(\mathbf{s}_t, \mathbf{a}_t)$, the maximum is soft. Hence, we can refer to $V$ and $Q$ as soft value functions and Q-functions, respectively. We can also consider the backup in Equation (6) in log-space. In the case of deterministic dynamics, this backup is given by

$$Q(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t, \mathbf{a}_t) + V(\mathbf{s}_{t+1}),$$

which exactly corresponds to the Bellman backup. However, when the dynamics are stochastic, the backup is given by

$$Q(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t, \mathbf{a}_t) + \log E_{\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)}[\exp(V(\mathbf{s}_{t+1}))]. \tag{7}$$

This backup is peculiar, since it does not consider the expected value at the next state, but a "soft max" over the next expected value. Intuitively, this produces Q-functions that are optimistic: if among the possible outcomes for the next state there is one outcome with a very high value, it will dominate the backup, even when there are other possible states that might be likely and have extremely low value. This creates risk seeking behavior: if an agent behaves according to this Q-function, it might take actions that have extremely high risk, so long as they have some non-zero probability of a high reward. Clearly, this behavior is not desirable in many cases, and the standard PGM described in this section is often not well suited to stochastic dynamics. In Section 3, we will describe a simple modification that makes the backup correspond to the soft Bellman backup in the case of stochastic dynamics also, by using the framework of variational inference.

5

## 2.4 Which Objective does This Inference Procedure Optimize?

In the previous section, we derived an inference procedure that can be used to obtain the distribution over actions conditioned on all of the optimality variables, $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T})$. But which objective does this policy actually optimize? Recall that the overall distribution is given by

$$p(\tau) = \left[p(\mathbf{s}_1)\prod_{t=1}^{T}p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)\right]\exp\left(\sum_{t=1}^{T}r(\mathbf{s}_t, \mathbf{a}_t)\right), \tag{8}$$

which we can simplify in the case of deterministic dynamics into Equation (5). In this case, the conditional distributions $p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T})$ are simply obtained by marginalizing the full trajectory distribution and conditioning the policy at each time step on $\mathbf{s}_t$. We can adopt an optimization-based approximate inference approach to this problem, in which case the goal is to fit an approximation $\pi(\mathbf{a}_t|\mathbf{s}_t)$ such that the trajectory distribution

$$\hat{p}(\tau) \propto \mathbb{1}[p(\tau) \neq 0]\prod_{t=1}^{T}\pi(\mathbf{a}_t|\mathbf{s}_t)$$

matches the distribution in Equation (5). In the case of exact inference, as derived in the previous section, the match is exact, which means that $D_{\mathrm{KL}}(\hat{p}(\tau)\|p(\tau)) = 0$, where $D_{\mathrm{KL}}$ is the KL-divergence. We can therefore view the inference process as minimizing $D_{\mathrm{KL}}(\hat{p}(\tau)\|p(\tau))$, which is given by

$$D_{\mathrm{KL}}(\hat{p}(\tau)\|p(\tau)) = -E_{\tau\sim\hat{p}(\tau)}[\log p(\tau) - \log\hat{p}(\tau)].$$

Negating both sides and substituting in the equations for $p(\tau)$ and $\hat{p}(\tau)$, we get

$$-D_{\mathrm{KL}}(\hat{p}(\tau)\|p(\tau)) = E_{\tau\sim\hat{p}(\tau)}\left[\log p(\mathbf{s}_1) + \sum_{t=1}^{T}(\log p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) + r(\mathbf{s}_t, \mathbf{a}_t)) - \right.$$
$$\left.\log p(\mathbf{s}_1) - \sum_{t=1}^{T}(\log p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) + \log\pi(\mathbf{a}_t|\mathbf{s}_t))\right]$$
$$= E_{\tau\sim\hat{p}(\tau)}\left[\sum_{t=1}^{T}r(\mathbf{s}_t, \mathbf{a}_t) - \log\pi(\mathbf{a}_t|\mathbf{s}_t)\right]$$
$$= \sum_{t=1}^{T}E_{(\mathbf{s}_t,\mathbf{a}_t)\sim\hat{p}(\mathbf{s}_t,\mathbf{a}_t))}[r(\mathbf{s}_t, \mathbf{a}_t) - \log\pi(\mathbf{a}_t|\mathbf{s}_t)]$$
$$= \sum_{t=1}^{T}E_{(\mathbf{s}_t,\mathbf{a}_t)\sim\hat{p}(\mathbf{s}_t,\mathbf{a}_t))}[r(\mathbf{s}_t, \mathbf{a}_t)] + E_{\mathbf{s}_t\sim\hat{p}(\mathbf{s}_t)}[\mathcal{H}(\pi(\mathbf{a}_t|\mathbf{s}_t))].$$

Therefore, minimizing the KL-divergence corresponds to maximizing the expected reward *and* the expected conditional entropy, in contrast to the standard control objective in Equation (1), which only maximizes reward. Hence, this type of control objective is sometimes referred to as maximum entropy reinforcement learning or maximum entropy control.

However, that in the case of stochastic dynamics, the solution is not quite so simple. Under stochastic dynamics, the optimized distribution is given by

$$\hat{p}(\tau) = p(\mathbf{s}_1|\mathcal{O}_{1:T})\prod_{t=1}^{T}p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t, \mathcal{O}_{1:T})p(\mathbf{a}_t|\mathbf{s}_t, \mathcal{O}_{1:T}), \tag{9}$$

where the initial state distribution and the dynamics are *also* conditioned on optimality. As a result of this, the dynamics and initial state terms in the KL-divergence do *not* cancel, and the objective does not have the simple entropy maximizing form derived above.[2] We can still fall back on the original KL-divergence minimization at the trajectory level, and write the objective as

$$-D_{\mathrm{KL}}(\hat{p}(\tau)\|p(\tau)) = E_{\tau\sim\hat{p}(\tau)}\left[\log p(\mathbf{s}_1) + \sum_{t=1}^{T}r(\mathbf{s}_t, \mathbf{a}_t) + \log p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)\right] + \mathcal{H}(\hat{p}(\tau)). \tag{10}$$

---

[2]In the deterministic case, we know that $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t, \mathcal{O}_{1:T}) = p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, since exactly one transition is ever possible.

However, because of the $\log p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ terms, this objective is difficult to optimize in a model-free setting. As discussed in the previous section, it also results in an optimistic policy that assumes a degree of control over the dynamics that is unrealistic in most control problems. In Section 3, we will derive a variational inference procedure that *does* reduce to the convenient objective in Equation (9) even in the case of stochastic dynamics, and in the process also addresses the risk-seeking behavior discussed in Section 2.3.

## 2.5 Alternative Model Formulations

It's worth pointing out that the definition of $p(\mathcal{O}_t = 1|\mathbf{s}_t, \mathbf{a}_t)$ in Equation (3) requires an additional assumption, which is that the rewards $r(\mathbf{s}_t, \mathbf{a}_t)$ are always negative.[3] Otherwise, we end up with a negative probability for $p(\mathcal{O}_t = 0|\mathbf{s}_t, \mathbf{a}_t)$. However, this assumption is not actually required: it's quite possible to instead define the graphical model with an undirected factor on $(\mathbf{s}_t, \mathbf{a}_t, \mathcal{O}_t)$, with an unnormalized potential given by $\Phi_t(\mathbf{s}_t, \mathbf{a}_t, \mathcal{O}_t) = \mathbb{1}_{\mathcal{O}_t=1} \exp(r(\mathbf{s}_t, \mathbf{a}_t))$. The potential for $\mathcal{O}_t = 0$ doesn't matter, since we always condition on $\mathcal{O}_t = 1$. This leads to the same exact inference procedure as the one we described above, but without the negative reward assumption. Once we are content to working with undirected graphical models, we can even remove the variables $\mathcal{O}_t$ completely, and simply add an undirected factor on $(\mathbf{s}_t, \mathbf{a}_t)$ with the potential $\Phi_t(\mathbf{s}_t, \mathbf{a}_t) = \exp(r(\mathbf{s}_t, \mathbf{a}_t))$, which is mathematically equivalent. This is the conditional random field formulation described by Ziebart (Ziebart, 2010). The analysis and inference methods in this model are identical to the ones for the directed model with explicit optimality variables $\mathcal{O}_t$, and the particular choice of model is simply a notational convenience. We will use the variables $\mathcal{O}_t$ in this article for clarity of derivation and stay within the directed graphical model framework, but all derivations are straightforward to reproduce in the conditional random field formulation.

Another common modification to this framework is to incorporate an explicit temperature $\alpha$ into the CPD for $\mathcal{O}_t$, such that $p(\mathcal{O}_t|\mathbf{s}_t, ba_t) = \exp(\frac{1}{\alpha}r(\mathbf{s}_t, \mathbf{a}_t))$. The corresponding maximum entropy objective can then be written equivalently as the expectation of the (original) reward, with an additional multiplier of $\alpha$ on the entropy term. This provides a natural mechanism to interpolate between entropy maximization and standard optimal control or RL: as $\alpha \to 0$, the optimal solution approaches the standard optimal control solution. Note that this does not actually increase the generality of the method, since the constant $\frac{1}{\alpha}$ can always be multiplied into the reward, but making this temperature constant explicit can help to illuminate the connection between standard and entropy maximizing optimal control.

Finally, it is worth remarking again on the role of discount factors: it is very common in reinforcement learning to use a Bellman backup of the form

$$Q(\mathbf{s}_t, \mathbf{a}_t) \leftarrow r(\mathbf{s}_t, \mathbf{a}_t) + \gamma E_{\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)}[V(\mathbf{s}_{t+1})],$$

where $\gamma \in (0, 1]$ is a discount factor. This allows for learning value functions in infinite-horizon settings, where the backup would otherwise be non-convergent for $\gamma = 1$, and reduces variance for Monte Carlo advantage estimators in policy gradient algorithms (Schulman et al., 2016). The discount factor can be viewed a simple redefinition of the system dynamics. If the initial dynamics are given by $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, adding a discount factor is equivalent to undiscounted value fitting under the modified dynamics $\bar{p}(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) = \gamma p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, where there is an additional transition with probability $1 - \gamma$, regardless of action, into an absorbing state with reward zero. We will omit $\gamma$ from the derivations in this article, but it can be inserted trivially in all cases simply by modifying the (soft) Bellman backups in any place where the expectation over $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ occurs, such as Equation (7) previously or Equation (15) in the next section.

## 3 Variational Inference and Stochastic Dynamics

The problematic nature of the maximum entropy framework in the case of stochastic dynamics, discussed in Section 2.3 and Section 2.4, in essence amounts to an assumption that the agent is allowed to control both its actions and the dynamics of the system in order to produce optimal trajectories, but its authority over the dynamics is penalized based on deviation from the true dynamics. Hence, the $\log p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ terms in Equation (10) can be factored out of the equations, producing additive

---

[3]This assumption is not actually very strong: if we assume the reward is bounded above, we can always construct an exactly equivalent reward simply by subtracting the maximum reward.

terms that corresponds to the cross-entropy between the posterior dynamics $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t, \mathcal{O}_{1:T})$ and the true dynamics $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$. This explains the risk-seeking nature of the method discussed in Section 2.3: if the agent is allowed to influence its dynamics, even a little bit, it would reasonably choose to remove unlikely but extremely bad outcomes of risky actions.

Of course, in practical reinforcement learning and control problems, such manipulation of system dynamics is not possible, and the resulting policies can lead to disastrously bad outcomes. We can correct this issue by modifying the inference procedure. In this section, we will derive this correction by freezing the system dynamics, writing down the corresponding maximum entropy objective, and deriving a dynamic programming procedure for optimizing it. Then we will show that this procedure amounts to a direct application of structured variational inference.

## 3.1 Maximum Entropy Reinforcement Learning with Fixed Dynamics

The issue discussed in Section 2.4 for stochastic dynamics can briefly be summarized as following: since the posterior dynamics distribution $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t, \mathcal{O}_{1:T})$ does not necessarily match the true dynamics $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, the agent assumes that it can influence the dynamics to a limited extent. A simple fix to this issue is to explicitly disallow this control, by forcing the posterior dynamics and initial state distributions to match $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ and $p(\mathbf{s}_1)$, respectively. Then, the optimized trajectory distribution is given simply by

$$\hat{p}(\tau) = p(\mathbf{s}_1) \prod_{t=1}^{T} p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)\pi(\mathbf{a}_t|\mathbf{s}_t),$$

and the same derivation as the one presented in Section 2.4 for the deterministic case results in the following objective:

$$-D_{\text{KL}}(\hat{p}(\tau)\|p(\tau)) = \sum_{t=1}^{T} E_{(\mathbf{s}_t, \mathbf{a}_t) \sim \hat{p}(\mathbf{s}_t, \mathbf{a}_t)}[r(\mathbf{s}_t, \mathbf{a}_t) + \mathcal{H}(\pi(\mathbf{a}_t|\mathbf{s}_t))]. \qquad (11)$$

That is, the objective is still to maximize reward and entropy, but now under stochastic transition dynamics. To optimize this objective, we can compute backward messages like we did in Section 2.3. However, since we are now starting from the maximization of the objective in Equation (11), we have to derive these backward messages from an optimization perspective as a dynamic programming algorithm. As before, we will begin with the base case of optimizing $\pi(\mathbf{a}_T|\mathbf{s}_T)$, which maximizes

$$E_{(\mathbf{s}_T, \mathbf{a}_T) \sim \hat{p}(\mathbf{s}_T, \mathbf{a}_T)}[r(\mathbf{s}_T, \mathbf{a}_T) - \log \pi(\mathbf{a}_T|\mathbf{s}_T)] =$$
$$E_{\mathbf{s}_T \sim \hat{p}(\mathbf{s}_T)}\left[ -D_{\text{KL}}\left( \pi(\mathbf{a}_T|\mathbf{s}_T)\|\frac{1}{\exp(V(\mathbf{s}_T))}\exp(r(\mathbf{s}_T, \mathbf{a}_T)) \right) + V(\mathbf{s}_T) \right], \qquad (12)$$

where the equality holds from the definition of KL-divergence, and $\exp(V(\mathbf{s}_T))$ is the normalizing constant for $\exp(r(\mathbf{s}_T, \mathbf{a}_T))$ with respect to $\mathbf{a}_T$ where $V(\mathbf{s}_T) = \log \int_{\mathcal{A}} \exp(r(\mathbf{s}_T, \mathbf{a}_T))d\mathbf{a}_T$, which is the same soft maximization as in Section 2.3. Since we know that the KL-divergence is minimized when the two arguments represent the same distribution, the optimal policy is given by

$$\pi(\mathbf{a}_T|\mathbf{s}_T) = \exp\left(r(\mathbf{s}_T, \mathbf{a}_T) - V(\mathbf{s}_T)\right), \qquad (13)$$

The recursive case can then computed as following: for a given time step $t$, $\pi(\mathbf{a}_t|\mathbf{s}_t)$ must maximize two terms:

$$E_{(\mathbf{s}_t, \mathbf{a}_t) \sim \hat{p}(\mathbf{s}_t, \mathbf{a}_t)}[r(\mathbf{s}_t, \mathbf{a}_t) - \log \pi(\mathbf{a}_t|\mathbf{s}_t)] + E_{(\mathbf{s}_t, \mathbf{a}_t) \sim \hat{p}(\mathbf{s}_t, \mathbf{a}_t)}[E_{\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)}[V(\mathbf{s}_{t+1})]]. \quad (14)$$

The first term follows directly from the objective in Equation (11), while the second term represents the contribution of $\pi(\mathbf{a}_t|\mathbf{s}_t)$ to the expectations of all subsequent time steps. The second term deserves a more in-depth derivation. First, consider the base case: given the equation for $\pi(\mathbf{a}_T|\mathbf{s}_T)$ in Equation (13), we can evaluate the objective for the policy by directly substituting this equation into Equation (12). Since the KL-divergence then evaluates to zero, we are left only with the $V(\mathbf{s}_T)$ term. In the recursive case, we note that we can rewrite the objective in Equation (14) as

$$E_{(\mathbf{s}_t, \mathbf{a}_t) \sim \hat{p}(\mathbf{s}_t, \mathbf{a}_t)}[r(\mathbf{s}_t, \mathbf{a}_t) - \log \pi(\mathbf{a}_t|\mathbf{s}_t)] + E_{(\mathbf{s}_t, \mathbf{a}_t) \sim \hat{p}(\mathbf{s}_t, \mathbf{a}_t)}[E_{\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)}[V(\mathbf{s}_{t+1})]] =$$
$$E_{\mathbf{s}_t \sim \hat{p}(\mathbf{s}_t)}\left[ -D_{\text{KL}}\left( \pi(\mathbf{a}_t|\mathbf{s}_t)\|\frac{1}{\exp(V(\mathbf{s}_t))}\exp(Q(\mathbf{s}_t, \mathbf{a}_t)) \right) + V(\mathbf{s}_t) \right],$$

8

where we now define

$$Q(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t, \mathbf{a}_t) + E_{\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)}[V(\mathbf{s}_{t+1})] \tag{15}$$

$$V(\mathbf{s}_t) = \log \int_{\mathcal{A}} \exp(Q(\mathbf{s}_t, \mathbf{a}_t)) d\mathbf{a}_t,$$

which corresponds to a standard Bellman backup with a soft maximization for the value function. Choosing

$$\pi(\mathbf{a}_t|\mathbf{s}_t) = \exp\left(Q(\mathbf{s}_t, \mathbf{a}_t) - V(\mathbf{s}_t)\right), \tag{16}$$

we again see that the KL-divergence evaluates to zero, leaving $E_{\mathbf{s}_t \sim \hat{p}(\mathbf{s}_t)}[V(\mathbf{s}_t)]$ as the only remaining term in the objective for time step $t$, just like in the base case of $t = T$. This means that, if we fix the dynamics and initial state distribution, and only allow the policy to change, we recover a Bellman backup operator that uses the expected value of the next state, rather than the optimistic estimate we saw in Section 2.3 (compare Equation (15) to Equation (7)). While this provides a solution to the practical problem of risk-seeking policies, it is perhaps a bit unsatisfying in its divergence from the convenient framework of probabilistic graphical models. In the next section, we will discuss how this procedure amounts to a direct application of structured variational inference.

### 3.2  Connection to Structured Variational Inference

One way to interpret the optimization procedure in Section 3.1 is as a particular type of structured variational inference. In structured variational inference, our goal is to approximate some distribution $p(\mathbf{y})$ with another, potentially simpler distribution $q(\mathbf{y})$. Typically, $q(\mathbf{y})$ is taken to be some tractable factorized distribution, such as a product of conditional distributions connected in a chain or tree, which lends itself to tractable exact inference. In our case, we aim to approximate $p(\tau)$, given by

$$p(\tau) = \left[p(\mathbf{s}_1) \prod_{t=1}^{T} p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)\right] \exp\left(\sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t)\right), \tag{17}$$

via the distribution

$$q(\tau) = q(\mathbf{s}_1) \prod_{t=1}^{T} q(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) q(\mathbf{a}_t|\mathbf{s}_t). \tag{18}$$

If we fix $q(\mathbf{s}_1) = p(\mathbf{s}_1)$ and $q(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) = p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, then $q(\tau)$ is exactly the distribution $\hat{p}(\tau)$ from Section 3.1, which we've renamed here to $q(\tau)$ to emphasize the connection to structured variational inference. Note that we've also renamed $\pi(\mathbf{a}_t|\mathbf{s}_t)$ to $q(\mathbf{a}_t|\mathbf{s}_t)$ for the same reason. In structured variational inference, approximate inference is performed by optimizing the variational lower bound (also called the evidence lower bound). Recall that our evidence here is that $\mathcal{O}_t = 1$ for all $t \in \{1, \ldots, T\}$, and the posterior is conditioned on the initial state $\mathbf{s}_1$. The variational lower bound is given by

$$\log p(\mathcal{O}_{1:T}) = \log \int \int p(\mathcal{O}_{1:T}, \mathbf{s}_{1:T}, \mathbf{a}_{1:T}) d\mathbf{s}_{1:T} d\mathbf{a}_{1:T}$$

$$= \log \int \int p(\mathcal{O}_{1:T}, \mathbf{s}_{1:T}, \mathbf{a}_{1:T}) \frac{q(\mathbf{s}_{1:T}, \mathbf{a}_{1:T})}{q(\mathbf{s}_{1:T}, \mathbf{a}_{1:T})} d\mathbf{s}_{1:T} d\mathbf{a}_{1:T}$$

$$= \log E_{(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}) \sim q(\mathbf{s}_{1:T}, \mathbf{a}_{1:T})} \left[\frac{p(\mathcal{O}_{1:T}, \mathbf{s}_{1:T}, \mathbf{a}_{1:T})}{q(\mathbf{s}_{1:T}, \mathbf{a}_{1:T})}\right]$$

$$\geq E_{(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}) \sim q(\mathbf{s}_{1:T}, \mathbf{a}_{1:T})} \left[\log p(\mathcal{O}_{1:T}, \mathbf{s}_{1:T}, \mathbf{a}_{1:T}) - \log q(\mathbf{s}_{1:T}, \mathbf{a}_{1:T})\right],$$

where the inequality on the last line is obtained via Jensen's inequality. Substituting the definitions of $p(\tau)$ and $q(\tau)$ from Equations (17) and (18), and noting the cancellation due to $q(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t) = p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, the bound reduces to

$$\log p(\mathcal{O}_{1:T}) \geq E_{(\mathbf{s}_{1:T}, \mathbf{a}_{1:T}) \sim q(\mathbf{s}_{1:T}, \mathbf{a}_{1:T})} \left[\sum_{t=1}^{T} r(\mathbf{s}_t, \mathbf{a}_t) - \log q(\mathbf{a}_t|\mathbf{s}_t)\right], \tag{19}$$

up to an additive constant. Optimizing this objective with respect to the policy $q(\mathbf{a}_t|\mathbf{s}_t)$ corresponds exactly to the objective in Equation (11). Intuitively, this means that this objective attempts to find

9

the closest match to the maximum entropy trajectory distribution, subject to the constraint that the agent is only allowed to modify the policy, and not the dynamics. Note that this framework can also easily accommodate any other structural constraints on the policy, including restriction to a particular distribution class (e.g., conditional Gaussian, or a categorical distribution parameterized by a neural network), or restriction to partial observability, where the entire state $\mathbf{s}_t$ is not available as an input, but rather the policy only has access to some non-invertible function of the state.

# 4   Approximate Inference with Function Approximation

We saw in the discussion above that a dynamic programming backward algorithm with updates that resemble Bellman backups can recover "soft" analogues of the value function and Q-function in the maximum entropy reinforcement learning framework, and the stochastic optimal policy can be recovered from the Q-function and value function. In this section, we will discuss how practical algorithms for high-dimensional or continuous reinforcement learning problems can be derived from this theoretical framework, with the use of function approximation. This will give rise to several prototypical methods that mirror corresponding techniques in standard reinforcement learning: policy gradients, actor-critic algorithms, and Q-learning.

## 4.1   Maximum Entropy Policy Gradients

One approach to performing structured variational inference is to directly optimize the evidence lower bound with respect to the variational distribution (Koller and Friedman, 2009). This approach can be directly applied to maximum entropy reinforcement learning. Note that the variational distribution consists of three terms: $q(\mathbf{s}_1)$, $q(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, and $q(\mathbf{a}_t|\mathbf{s}_t)$. The first two terms are fixed to $p(\mathbf{s}_1)$ and $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, respectively, leaving only $q(\mathbf{a}_t|\mathbf{s}_t)$ to vary. We can parameterize this distribution with any expressive conditional, with parameters $\theta$, and will therefore denote it as $q_\theta(\mathbf{a}_t|\mathbf{s}_t)$. The parameters could correspond, for example, to the weights in a deep neural network, which takes $\mathbf{s}_t$ as input and outputs the parameters of some distribution class. In the case of discrete actions, the network could directly output the parameters of a categorical distribution (e.g., via a soft max operator). In the case of continuous actions, the network could output the parameters of an exponential family distribution, such as a Gaussian. In all cases, we can directly optimize the objective in Equation (11) by estimating its gradient using samples. This gradient has a form that is nearly identical to the standard policy gradient (Williams, 1992), which we summarize here for completeness. First, let us restate the objective as following:

$$J(\theta) = \sum_{t=1}^{T} E_{(\mathbf{s}_t, \mathbf{a}_t) \sim q(\mathbf{s}_t, \mathbf{a}_t)} \left[ r(\mathbf{s}_t, \mathbf{a}_t) - \mathcal{H}(q_\theta(\mathbf{a}_t|\mathbf{s}_t)) \right].$$

The gradient is then given by

$$\nabla_\theta J(\theta) = \sum_{t=1}^{T} \nabla_\theta E_{(\mathbf{s}_t, \mathbf{a}_t) \sim q(\mathbf{s}_t, \mathbf{a}_t)} \left[ r(\mathbf{s}_t, \mathbf{a}_t) + \mathcal{H}(q_\theta(\mathbf{a}_t|\mathbf{s}_t)) \right]$$

$$= \sum_{t=1}^{T} E_{(\mathbf{s}_t, \mathbf{a}_t) \sim q(\mathbf{s}_t, \mathbf{a}_t)} \left[ \nabla_\theta \log q_\theta(\mathbf{a}_t|\mathbf{s}_t) \left( \sum_{t'=t}^{T} r(\mathbf{s}_{t'}, \mathbf{a}_{t'}) - \log q_\theta(\mathbf{a}_{t'}|\mathbf{s}_{t'}) - 1 \right) \right]$$

$$= \sum_{t=1}^{T} E_{(\mathbf{s}_t, \mathbf{a}_t) \sim q(\mathbf{s}_t, \mathbf{a}_t)} \left[ \nabla_\theta \log q_\theta(\mathbf{a}_t|\mathbf{s}_t) \left( \sum_{t'=t}^{T} r(\mathbf{s}_{t'}, \mathbf{a}_{t'}) - \log q_\theta(\mathbf{a}_{t'}|\mathbf{s}_{t'}) - b(\mathbf{s}_{t'}) \right) \right],$$

where the second line follows from applying the likelihood ratio trick (Williams, 1992) and the definition of entropy to obtain the $\log q_\theta(\mathbf{a}_{t'}|\mathbf{s}_{t'})$ term. The $-1$ comes from the derivative of the entropy term. The last line follows by noting that the gradient estimator is invariant to additive state-dependent constants, and replacing $-1$ with a state-dependent baseline $b(\mathbf{s}_{t'})$. The resulting policy gradient estimator exactly matches a standard policy gradient estimator, with the only modification being the addition of the $-\log q_\theta(\mathbf{a}_{t'}|\mathbf{s}_{t'})$ term to the reward at each time step $t'$. Intuitively, the reward of each action is modified by subtracting the log-probability of that action under the current policy, which causes the policy to maximize entropy. This gradient estimator can be written more

compactly as

$$\nabla_\theta J(\theta) = \sum_{t=1}^{T} E_{(\mathbf{s}_t,\mathbf{a}_t)\sim q(\mathbf{s}_t,\mathbf{a}_t)} \left[ \nabla_\theta \log q_\theta(\mathbf{a}_t|\mathbf{s}_t)\hat{A}(\mathbf{s}_t,\mathbf{a}_t) \right],$$

where $\hat{A}(\mathbf{s}_t,\mathbf{a}_t)$ is an advantage estimator. Any standard advantage estimator, such as the GAE estimator (Schulman et al., 2016), can be used in place of the standard baselined Monte Carlo return above. Again, the only necessary modification is to add $-\log q_\theta(\mathbf{a}_{t'}|\mathbf{s}_{t'})$ to the reward at each time step $t'$. As with standard policy gradients, a practical implementation of this method estimates the expectation by sampling trajectories from the current policy, and may be improved by following the natural gradient direction.

## 4.2 Maximum Entropy Actor-Critic Algorithms

Instead of directly differentiating the variational lower bound, we can adopt a message passing approach which, as we will see later, can produce lower-variance gradient estimates. First, note that we can write down the following equation for the optimal target distribution for $q(\mathbf{a}_t|\mathbf{s}_t)$:

$$q^\star(\mathbf{a}_t|\mathbf{s}_t) = \frac{1}{Z}\exp\left(E_{q(\mathbf{s}_{(t+1):T},\mathbf{a}_{(t+1):T}|\mathbf{s}_t,\mathbf{a}_t)}\left[\sum_{t'=t}^{T}\log p(\mathcal{O}_{t'}|\mathbf{s}_{t'},\mathbf{a}_{t'}) - \sum_{t'=t+1}^{T}\log q(\mathbf{a}_{t'}|\mathbf{s}_{t'})\right]\right).$$

This is because conditioning on $\mathbf{s}_t$ makes the action $\mathbf{a}_t$ completely independent of all past states, but the action still depends on all future states and actions. Note that the dynamics terms $p(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)$ and $q(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)$ do not appear in the above equation, since they perfectly cancel. We can simplify the expectation above as follows:

$$E_{q(\mathbf{s}_{(t+1):T},\mathbf{a}_{(t+1):T}|\mathbf{s}_t,\mathbf{a}_t)}[\log p(\mathcal{O}_{t:T}|\mathbf{s}_{t:T},\mathbf{a}_{t:T})] =$$

$$\log p(\mathcal{O}_t|\mathbf{s}_t,\mathbf{a}_t) + E_{q(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)}\left[E\left[\sum_{t'=t+1}^{T}\log p(\mathcal{O}_{t'}|\mathbf{s}_{t'},\mathbf{a}_{t'}) - \log q(\mathbf{a}_{t'}|\mathbf{s}_{t'})\right]\right].$$

In this case, note that the inner expectation does not contain $\mathbf{s}_t$ or $\mathbf{a}_t$, and therefore makes for a natural representation for a message that can be sent from future states. We will denote this message $V(\mathbf{s}_{t+1})$, since it will correspond to a soft value function:

$$V(\mathbf{s}_t) = E\left[\sum_{t'=t+1}^{T}\log p(\mathcal{O}_{t'}|\mathbf{s}_{t'},\mathbf{a}_{t'}) - \log q(\mathbf{a}_{t'}|\mathbf{s}_{t'})\right]$$

$$= E_{q(\mathbf{a}_t|\mathbf{s}_t)}[\log p(\mathcal{O}_t|\mathbf{s}_t,\mathbf{a}_t) - \log q(\mathbf{a}_t|\mathbf{s}_t) + E_{q(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)}[V(\mathbf{s}_{t+1})]].$$

For convenience, we can also define a Q-function as

$$Q(\mathbf{s}_t,\mathbf{a}_t) = \log p(\mathcal{O}_t|\mathbf{s}_t,\mathbf{a}_t) + E_{q(\mathbf{s}_{t+1}|\mathbf{s}_t,\mathbf{a}_t)}[V(\mathbf{s}_{t+1})],$$

such that $V(\mathbf{s}_t) = E_{q(\mathbf{a}_t|\mathbf{s}_t)}[Q(\mathbf{s}_t,\mathbf{a}_t) - \log q(\mathbf{a}_t|\mathbf{s}_t)]$, and the optimal policy is

$$q^\star(\mathbf{a}_t|\mathbf{s}_t) = \frac{\exp(Q(\mathbf{s}_t,\mathbf{a}_t))}{\log\int_\mathcal{A}\exp(Q(\mathbf{s}_t,\mathbf{a}_t))d\mathbf{a}_t}. \tag{20}$$

Note that, in this case, the value function and Q-function correspond to the values of the current policy $q(\mathbf{a}_t|\mathbf{s}_t)$, rather than the optimal value function and Q-function, as in the case of dynamic programming. However, at convergence, when $q(\mathbf{a}_t|\mathbf{s}_t) = q^\star(\mathbf{a}_t|\mathbf{s}_t)$ for each $t$, we have

$$V(\mathbf{s}_t) = E_{q(\mathbf{a}_t|\mathbf{s}_t)}[Q(\mathbf{s}_t,\mathbf{a}_t) - \log q(\mathbf{a}_t|\mathbf{s}_t)]$$

$$= E_{q(\mathbf{a}_t|\mathbf{s}_t)}\left[Q(\mathbf{s}_t,\mathbf{a}_t) - Q(\mathbf{s}_t,\mathbf{a}_t) + \log\int_\mathcal{A}\exp(Q(\mathbf{s}_t,\mathbf{a}_t))d\mathbf{a}_t\right]$$

$$= \log\int_\mathcal{A}\exp(Q(\mathbf{s}_t,\mathbf{a}_t))d\mathbf{a}_t, \tag{21}$$

which is the familiar soft maximum from Section 2.3. We now see that the optimal variational distribution for $q(\mathbf{a}_t|\mathbf{s}_t)$ can be computed by passing messages backward through time, and the messages are given by $V(\mathbf{s}_t)$ and $Q(\mathbf{s}_t,\mathbf{a}_t)$.

So far, this derivation assumes that the policy and messages can be represented exactly. We can relax the first assumption in the same way as in the preceding section. We first write down the variational lower bound for a single factor $q(\mathbf{a}_t|\mathbf{s}_t)$ as following:

$$\max_{q(\mathbf{a}_t|\mathbf{s}_t)} E_{\mathbf{s}_t \sim q(\mathbf{s}_t)} \left[ E_{\mathbf{a}_t \sim q(\mathbf{a}_t|\mathbf{s}_t)}[Q(\mathbf{s}_t, \mathbf{a}_t) - \log q(\mathbf{a}_t|\mathbf{s}_t)] \right]. \tag{22}$$

It's straightforward to show that this objective is simply the full variational lower bound, which is given by $E_{q(\tau)}[\log p(\tau) - \log q(\tau)]$, restricted to just the terms that include $q(\mathbf{a}_t|\mathbf{s}_t)$. If we restrict the class of policies $q(\mathbf{a}_t|\mathbf{s}_t)$ so that they cannot represent $q^\star(\mathbf{a}_t|\mathbf{s}_t)$ exactly, we can still optimize the objective in Equation (22) by computing its gradient, which is given by

$$E_{\mathbf{s}_t \sim q(\mathbf{s}_t)} \left[ E_{\mathbf{a}_t \sim q(\mathbf{a}_t|\mathbf{s}_t)}[\nabla \log q(\mathbf{a}_t|\mathbf{s}_t)(Q(\mathbf{s}_t, \mathbf{a}_t) - \log q(\mathbf{a}_t|\mathbf{s}_t) - b(\mathbf{s}_t))] \right],$$

where $b(\mathbf{s}_t)$ is any state-dependent baseline. This gradient can computed using samples from $q(\tau)$ and, like the policy gradient in the previous section, is directly analogous to a classic likelihood ratio policy gradient. The modification lies in the use of the backward message $Q(\mathbf{s}_t, \mathbf{a}_t)$ in place of the Monte Carlo advantage estimate. The algorithm therefore corresponds to an actor-critic algorithm, which generally provides lower variance gradient estimates.

In order to turn this into a practical algorithm, we must also be able to approximate the backward message $Q(\mathbf{s}_t, \mathbf{a}_t)$ and $V(\mathbf{s}_t)$. A simple and straightforward approach is to represent them with parameterized functions $Q_\phi(\mathbf{s}_t, \mathbf{a}_t)$ and $V_\psi(\mathbf{s}_t)$, with parameters $\phi$ and $\psi$, and optimize the parameters to minimize a squared error objectives

$$\mathcal{E}(\phi) = E_{(\mathbf{s}_t, \mathbf{a}_t) \sim q(\mathbf{s}_t, \mathbf{a}_t)} \left[ \left( r(\mathbf{s}_t, \mathbf{a}_t) + E_{q(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)}[V_\psi(\mathbf{s}_{t+1})] - Q_\phi(\mathbf{s}_t, \mathbf{a}_t) \right)^2 \right] \tag{23}$$

$$\mathcal{E}(\psi) = E_{\mathbf{s}_t \sim q(\mathbf{s}_t)} \left[ \left( E_{\mathbf{a}_t \sim q(\mathbf{a}_t|\mathbf{s}_t)}[Q_\phi(\mathbf{s}_t, \mathbf{a}_t) - \log q(\mathbf{a}_t|\mathbf{s}_t)] - V_\psi(\mathbf{s}_t, \mathbf{a}_t) \right)^2 \right].$$

This interpretation gives rise to a few interesting possibilities for maximum entropy actor-critic and policy iteration algorithms. First, it suggests that it may be beneficial to keep track of both $V(\mathbf{s}_t)$ and $Q(\mathbf{s}_t, \mathbf{a}_t)$ networks. This is perfectly reasonable in a message passing framework, and in practice might have many of the same benefits as the use of a target network, where the updates to $Q$ and $V$ can be staggered or damped for stability. Second, it suggests that policy iteration or actor-critic methods might be preferred (over, for example, direct Q-learning), since they explicitly handle both approximate messages and approximate factors in the structured variational approximation. This is precisely the scheme employed by the soft actor-critic algorithm (Haarnoja et al., 2018b).

## 4.3 Soft Q-Learning

We can derive an alternative form for a reinforcement learning algorithm without using an explicit policy parameterization, fitting only the messages $Q_\phi(\mathbf{s}_t, \mathbf{a}_t)$. In this case, we assume an implicit parameterization for both the value function $V(\mathbf{s}_t)$ and policy $q(\mathbf{a}_t|\mathbf{s}_t)$, where

$$V(\mathbf{s}_t) = \log \int_{\mathcal{A}} \exp(Q(\mathbf{s}_t, \mathbf{a}_t)) d\mathbf{a}_t,$$

as in Equation (21), and

$$q(\mathbf{a}_t|\mathbf{s}_t) = \exp(Q(\mathbf{s}_t, \mathbf{a}_t) - V(\mathbf{s}_t)),$$

which corresponds directly to Equation (20). In this case, no further parameterization is needed beyond $Q_\phi(\mathbf{s}_t, \mathbf{a}_t)$, which can be learned by minimizing the error in Equation (23), substituting the implicit equation for $V(\mathbf{s}_t)$ in place of $V_\psi(\mathbf{s}_t)$. We can write the resulting gradient update as

$$\phi \leftarrow \phi - \alpha E \left[ \frac{dQ_\phi}{d\phi}(\mathbf{s}_t, \mathbf{a}_t) \left( Q_\phi(\mathbf{s}_t, \mathbf{a}_t) - \left( r(\mathbf{s}_t, \mathbf{a}_t) + \log \int_{\mathcal{A}} \exp(Q(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})) d\mathbf{a}_{t+1} \right) \right) \right].$$

It is worth pointing out the similarity to the standard Q-learning update:

$$\phi \leftarrow \phi - \alpha E \left[ \frac{dQ_\phi}{d\phi}(\mathbf{s}_t, \mathbf{a}_t) \left( Q_\phi(\mathbf{s}_t, \mathbf{a}_t) - \left( r(\mathbf{s}_t, \mathbf{a}_t) + \max_{\mathbf{a}_{t+1}} Q_\phi(\mathbf{s}_{t+1}, \mathbf{a}_{t+1}) \right) \right) \right].$$

Where the standard Q-learning update has a $\max$ over $\mathbf{a}_{t+1}$, the soft Q-learning update has a "soft" max. As the magnitude of the reward increases, the soft update comes to resemble the hard update.

In the case of discrete actions, this update is straightforward to implement, since the integral is replaced with a summation, and the policy can be extracted simply by normalizing the Q-function. In the case of continuous actions, a further level of approximation is needed to evaluate the integral using samples. Sampling from the implicit policy is also non-trivial, and requires an approximate inference procedure, as discussed by Haarnoja et al. (Haarnoja et al., 2017).

We can further use this framework to illustrate an interesting connection between soft Q-learning and policy gradients. According to the definition of the policy in Equation (20), which is defined entirely in terms of $Q_\phi(\mathbf{s}_t, \mathbf{a}_t)$, we can derive an alternative gradient with respect to $\phi$ starting from the policy gradient. This derivation represents a connection between policy gradient and Q-learning that is not apparent in the standard framework, but becomes apparent in the maximum entropy framework. The full derivation is provided by Haarnoja et al. (Haarnoja et al., 2017) (Appendix B). The final gradient corresponds to

$$\nabla_\phi J(\phi) = \sum_{t=1}^{T} E_{(\mathbf{s}_t, \mathbf{a}_t) \sim q(\mathbf{s}_t, \mathbf{a}_t)} \left[ \left( \nabla_\phi Q(\mathbf{s}_t, \mathbf{a}_t) - \nabla_\phi V(\mathbf{s}_t) \right) \hat{A}(\mathbf{s}_t, \mathbf{a}_t) \right].$$

The soft Q-learning gradient can equivalently be written as

$$\nabla_\phi J(\phi) = \sum_{t=1}^{T} E_{(\mathbf{s}_t, \mathbf{a}_t) \sim q(\mathbf{s}_t, \mathbf{a}_t)} \left[ \nabla_\phi Q(\mathbf{s}_t, \mathbf{a}_t) \hat{A}(\mathbf{s}_t, \mathbf{a}_t) \right], \tag{24}$$

where we substitute the target value $r(\mathbf{s}_t, \mathbf{a}_t) + V(\mathbf{s}_{t+1})$ for $\hat{A}(\mathbf{s}_t, \mathbf{a}_t)$, taking advantage of the fact that we can use any state-dependent baseline. Although these gradients are not exactly equal, the extra term $-\nabla_\phi V(\mathbf{s}_t)$ simply accounts for the fact that the policy gradient alone is insufficient to resolve one extra degree of freedom in $Q(\mathbf{s}_t, \mathbf{a}_t)$: the addition or subtraction of an action-independent constant. We can eliminate this term if we add the policy gradient with respect to $\phi$ together with Bellman error minimization for $V(\mathbf{s}_t)$, which has the gradient

$$\nabla_\phi V(\mathbf{s}_t) E_{\mathbf{a}_t \sim q(\mathbf{a}_t | \mathbf{s}_t)} \left[ r(\mathbf{s}_t, \mathbf{a}_t) + E_{\mathbf{s}_{t+1} \sim q(\mathbf{s}_{t+1} | \mathbf{s}_t, \mathbf{a}_t)} [V(\mathbf{s}_{t+1})] \right] = \nabla_\phi V(\mathbf{s}_t) E_{\mathbf{a}_t \sim q(\mathbf{a}_t | \mathbf{s}_t)} [\hat{Q}(\mathbf{s}_t, \mathbf{a}_t)].$$

Noting that $\hat{Q}(\mathbf{s}_t, \mathbf{a}_t)$ is simply a (non-baselined) return estimate, we can show that the sum of the policy gradient and value gradient exactly matches Equation (24) for a particular choice of state-dependent baseline, since the term $\nabla_\phi V(\mathbf{s}_t) E_{\mathbf{a}_t \sim q(\mathbf{a}_t | \mathbf{s}_t)} [\hat{Q}(\mathbf{s}_t, \mathbf{a}_t)]$ cancels against the term $-\nabla_\phi V(\mathbf{s}_t) \hat{A}(\mathbf{s}_t, \mathbf{a}_t)$ in expectation over $\mathbf{a}_t$ when $\hat{A}(\mathbf{s}_t, \mathbf{a}_t) = \hat{Q}(\mathbf{s}_t, \mathbf{a}_t)$ (that is, when we use a baseline of zero). This completes the proof of a general equivalence between soft Q-learning and policy gradients.

## 5 Review of Prior Work

In this section, we will discuss a variety of prior works that have sought to explore the connection between inference and control, make use of this connection to devise more effective learning algorithms, and extend it into other applications, such as intent inference and human behavior forecasting. We will first discuss the variety of frameworks proposed in prior work that are either equivalent to the approach presented in this article, or special cases (or generalization) thereof (Section 5.1). We will then discuss alternative formulations that, though similar, differ in some critical way (Section 5.2). We will then discuss specific reinforcement learning algorithms that build on the maximum entropy framework (Section 5.3), and conclude with a discussion of applications of the maximum entropy framework in other areas, such as intent inference, human behavior modeling, and forecasting (Section 5.4).

### 5.1 Frameworks for Control as Inference

Framing control, decision making, and reinforcement learning as a probabilistic inference and learning problem has a long history, going back to original work by Rudolf Kalman (Kalman, 1960), who described how the Kalman smoothing algorithm can also be used to solve control problems with linear dynamics and quadratic costs (the "linear-quadratic regulator" or LQR setting). It is worth noting that, in the case of linear-quadratic systems, the maximum entropy solution is a linear-Gaussian policy where the mean corresponds exactly to the optimal *deterministic* policy. This sometimes referred to as the Kalman duality (Todorov, 2008). Unfortunately, this elegant duality does

not in general hold for non-LQR systems: the maximum entropy framework for control as inference generalizes standard optimal control, and the optimal stochastic policy for a non-zero temperature (see Section 2.5) does not in general have the optimal deterministic policy as its mean.

Subsequent work expanded further on the connection between control and inference. Attias (2003) proposed to implement a planning algorithm by means of the Baum-Welch-like method in HMM-style models. Todorov (2006) formulated a class of reinforcement learning problems termed "linearly-solvable" MDPs (LMDPs). LMDPs correspond to the graphical model described in Section 2, but immediately marginalize out the actions or, equivalently, posit that actions are equivalent to the next state, allowing the entire framework to operate entirely on states rather than actions. This gives rise to a simple and elegant framework that is especially tractable in the tabular setting. In the domain of optimal control, Kappen (2011) formulated a class of path integral control problems that also correspond to the graphical model discussed in Section 2, but derived starting from a continuous time formulation and formulated as a diffusion process. This continuous time generalization arrives at the same solution in discrete time, but requires considerably more stochastic processes machinery, so is not discussed in detail in this article. Similar work by Toussaint and colleagues (Toussaint, 2009; Rawlik et al., 2013) formulated graphical models for solving decision making problems, indirectly arriving at the same framework as in the previously discussed works. In the case of Toussaint (2009), expectation propagation (Minka, 2001) was adapted for approximate message passing during planning. Ziebart (2010) formulated learning in PGMs corresponding to the control or reinforcement learning problem as a problem of learning reward functions, and used this connection to derive maximum entropy *inverse* reinforcement learning algorithms, which are discussed in more detail in Section 5.4. Ziebart's derivation of the relationship between decision making, conditional random fields, and PGMs also provides a thorough exploration of the foundational theory in this field, and is a highly recommend compendium to accompany this article for readers seeking a more in-depth theoretical discussion and connections to maximum entropy models (Ziebart, 2010). The particular PGM studied by Ziebart is discussed in Section 2.5: although Ziebart frames the model as a conditional random field, without the auxiliary optimality variables $\mathcal{O}_t$, this formulation is equivalent to the one discussed here. Furthermore, the maximum causal entropy method discussed by Ziebart can be shown to be equivalent to the variational inference formulation presented in Section 3.1.

## 5.2 Related but Distinct Approaches

All of the methods discussed in the previous section are either special cases or generalizations of the control as inference framework presented in this article. A number of other works have presented related approaches that also aim to unify control and inference, but do so in somewhat different ways. We survey some of these prior techniques in this section, and describe their technical and practical differences from the presented formulation.

**Boltzmann Exploration.** The form of the optimal policy in the maximum entropy framework (e.g., Equation (20)) suggests a very natural exploration strategy: actions that have large Q-value should be taken more often, while actions that have low Q-value should be taken less often, and the stochastic exploration strategy has the form of a Boltzmann-like distribution, with the Q-function acting as the negative energy. A large number of prior methods (Sutton, 1990; Kaelbling et al., 1996) have proposed to use such a policy distribution as an exploration strategy, but in the context of a reinforcement learning algorithm where the Q-function is learned via the standard ("hard") max operator, corresponding to a temperature (see Section 2.5) of zero. Boltzmann exploration therefore does not optimize the maximum entropy objective, but rather serves as a heuristic modification to enable improved exploration. A closely related idea is presented in the work on energy-based reinforcement learning (Sallans and Hinton, 2004; Heess et al., 2013), where the free energy of an energy-based model (in that case, a restricted Boltzmann machine) is adjusted based on a reinforcement learning update rule, such that the energy corresponds to the negative Q-function. Interestingly, energy-based reinforcement learning can optimize either the maximum entropy objective or the standard objective (with Boltzmann exploration), based on the type of update rule that is used. When used with an on-policy SARSA update rule, as proposed by Sallans and Hinton (2004), the method actually does optimize the maximum entropy objective, since the policy uses the Boltzmann distribution. However, when updated using an off-policy Q-learning objective with a hard max, the method reduces to Boltzmann exploration and optimizes the standard RL objective.

**Entropy Regularization.** In the context of policy gradient and actor-critic methods, a commonly used technique is to use "entropy regularization," where an entropy maximization term is added to the policy objective to prevent the policy from becoming too deterministic prematurely. This technique was proposed as early as the first work on the REINFORCE algorithm (Williams and Peng, 1991; Williams, 1992), and is often used in recent methods (see, e.g., discussion by O'Donoghue et al. (2017)). While the particular technique for incorporating this entropy regularizer varies, typically the simplest way is to simply add the gradient of the policy entropy at each sampled state to a standard policy gradient estimate, which itself may use a critic. Note that this is *not*, in general, equivalent to the maximum entropy objective, which not only optimizes for a policy with maximum entropy, but also optimizes the policy itself to *visit* states where it has high entropy. Put another way, the maximum entropy objective optimizes the expectation of the entropy with respect to the policy's state distribution, while entropy regularization only optimizes the policy entropy at the states that are visited, without actually trying to modify the policy itself to visit high-entropy states (see, e.g., Equation (2) in O'Donoghue et al. (2017)). While this does correspond to a well-defined objective, that objective is rather involved to write out and generally not mentioned in work that uses entropy regularization. The technique is typically presented as a heuristic modification to the policy gradient. Interestingly, it is actually easier to perform proper maximum entropy RL than entropy regularization: maximum entropy RL with a policy gradient or actor-critic method only requires subtracting $\log \pi(\mathbf{a}|\mathbf{s})$ from the reward function, while heuristic entropy maximization typically uses an explicit entropy gradient.

**Variational Policy Search and Expectation Maximization.** Another formulation of the reinforcement learning problem with strong connections to probabilistic inference is the formulation of policy search in an expectation-maximization style algorithm. One common way to accomplish this is to directly treat rewards as a probability density, and then use a "pseudo-likelihood" written as

$$J(\theta) = \int r(\tau)p(\tau|\theta)d\tau,$$

where $r(\tau)$ is the total reward along a trajectory, and $p(\tau|\theta)$ is the probability of observing a trajectory $\tau$ given a policy parameter vector $\theta$. Assuming $r(\tau)$ is positive and bounded and applying Jensen's inequality results in a variety of algorithms (Peters and Schaal, 2007; Hachiya et al., 2009; Neumann, 2011; Abdolmaleki et al., 2018), including reward-weighted regression (Peters and Schaal, 2007), that all follow the following general recipe: samples are weighted according to some function of their return (potentially with importance weights), and the policy is then updated by optimizing a regression objective to match the sample actions, weighted by these weights. The result is that samples with higher return are matched more closely, while those with low return are ignored. Variational policy search methods also fall into this category (Neumann, 2011; Levine and Koltun, 2013b), sometimes with the modification of using explicit trajectory optimization rather than reweighting to construct the target actions (Levine and Koltun, 2013b), and sometimes using an exponential transformation on $r(\tau)$ to ensure positivity. Unlike the approach discussed in this article, the use of the reward as a "pseudo-likelihood" does not correspond directly to a well-defined probabilistic model, though the application of Jensen's inequality can still be motivated simply from the standpoint of deriving a bound for the RL optimization problem. A more serious disadvantage of this class of methods is that, by regressing onto the reweighted samples, the method loses the ability to properly handle risk for stochastic dynamics and policies. Consider, for example, a setting where we aim to fit a unimodal policy for a stateless problem with a 1D action. If we have a high reward for the action $-1$ and $+1$, and a low reward for the action $0$, the optimal fit will still place all of the probability mass in the middle, at the action $0$, which is the worst possible option. Mathematically, this problem steps from the fact that supervised learning matches a target distribution by minimizing a KL-divergence of the form $D_{\mathrm{KL}}(p_{\mathrm{tgt}}\|p_\theta)$, where $p_{\mathrm{tgt}}$ is the target distribution (e.g., the reward or exponentiated reward). RL instead minimizes a KL-divergence of the form $D_{\mathrm{KL}}(p_\theta\|p_{\mathrm{tgt}})$, which prioritizes finding a mode of the target distribution rather than matching its moments. This issue is discussed in more detail in Section 5.3.5 of (Levine, 2014). In general, the issue manifests itself as risk-seeking behavior, though distinct in nature from the risk-seeking behavior discussed in Section 2.4. Note that Toussaint and Storkey (2006) also propose an expectation-maximization based algorithm for control as inference, but in a framework that does in fact yield maximum expected reward solutions, with a similar formulation to the one in this article.

**KL-Divergence Constraints for Policy Search.** Policy search methods frequently employ a constraint between the new policy and the old policy at each iteration, in order to bound the change in the policy distribution and thereby ensure smooth, stable convergence. Since policies are distributions, a natural choice for the form of this constraint is a bound on the KL-divergence between the new policy and the old one (Bagnell and Schneider, 2003; Peters et al., 2010; Levine and Abbeel, 2014; Schulman et al., 2015; Abdolmaleki et al., 2018). When we write out the Lagrangian of the resulting optimization problem, we typically end up with a maximum entropy optimization problem similar to the one in Equation (11), where instead of taking the KL-divergence between the new policy and exponentiated reward, we instead have a KL-divergence between the new policy and the old one. This corresponds to a maximum entropy optimization where the reward is $r(\mathbf{s}, \mathbf{a}) + \lambda \log \bar{\pi}(\mathbf{a}|\mathbf{s})$, where $\lambda$ is the Lagrange multiplier and $\bar{\pi}$ is the old policy, and the entropy term has a weight of $\lambda$. This is equivalent to a maximum entropy optimization where the entropy has a weight of one, and the reward is scaled by $\frac{1}{\lambda}$. Thus, although none of these methods actually aim to optimize the maximum entropy objective in the end, each step of the policy update involves solving a maximum entropy problem. A similar approach is proposed by Rawlik et al. (2013), where a sequence of maximum entropy problems is solved in a Q-learning style framework to eventually arrive at the standard maximum reward solution.

## 5.3    Reinforcement Learning Algorithms

Maximum entropy reinforcement learning algorithms have been proposed in a range of frameworks and with a wide variety of assumptions. The path integral framework has been used to derive algorithms for both optimal control and planning (Kappen, 2011) and policy search via reinforcement learning (Theodorou et al., 2010). The framework of linearly solvable MDPs has been used to derive policy search algorithms (Todorov, 2010), value function based algorithms (Todorov, 2006), and inverse reinforcement learning algorithms (Dvijotham and Todorov, 2010). More recently, entropy maximization has been used as a component in algorithms based on model-free policy search with importance sampling and its variants (Levine and Koltun, 2013a; Nachum et al., 2017a,b), model-based algorithms based on the guided policy search framework (Levine and Abbeel, 2014; Levine and Koltun, 2014; Levine et al., 2016), and a variety of methods based on soft Q-learning (Haarnoja et al., 2017; Schulman et al., 2017) and soft actor-critic algorithms (Haarnoja et al., 2018b; Hausman et al., 2018).

The particular reasons for the use of the control as inference framework differ between each of these algorithms. The motivation for linearly solvable MDPs is typically based on computationally tractable exact solutions for tabular settings (Todorov, 2010), which are enabled essentially by dispensing with the non-linear maximization operator in the standard RL framework. Although the maximum entropy dynamic programming equations are still not linear in terms of value functions and Q-functions, they are linear under an exponential transformation. The reason for this is quite natural: since these methods implement sum-product message passing, the only operations in the original probability space are summations and multiplications. However, for larger problems where tabular representations are impractical, these benefits are not apparent.

In the case of path consistency methods (Nachum et al., 2017a,b), the maximum entropy framework offers an appealing mechanism for off-policy learning. In the case of guided policy search, it provides a natural method for matching distributions between model-based local policies and a global policy that unifies the local policy into a single globally coherent strategy (Levine and Koltun, 2014; Levine and Abbeel, 2014; Levine et al., 2016). For the more recent model-free maximum entropy algorithms, such as soft Q-learning (Haarnoja et al., 2017) and soft-actor critic (Haarnoja et al., 2018b), as well the work of Hausman et al. (2018), the benefits are improved stability and model-free RL performance, improved exploration, and the ability to pre-train policies for diverse and under-specified goals. For example, Haarnoja et al. (2017) present a quadrupedal robot locomotion task where the reward depends only on the speed of the robot's motion, regardless of direction. In a standard RL framework, this results in a policy that runs in an arbitrary direction. Under the maximum entropy framework, the optimal policy runs in all directions with equal probability. This makes it well-suited for pretraining general-purpose policies that can then be finetuned for more narrowly tailored tasks. More recently, Haarnoja et al. also showed that maximum entropy policies can be composed simply by adding their Q-functions, resulting in a Q-function with bounded difference against the optimal Q-function for the corresponding composed reward (Haarnjoa et al., 2018).

Recently, a number of papers have explored how the control as inference or maximum entropy reinforcement learning framework can be extended to add additional latent variables to the model, such that the policy is given by $\pi(\mathbf{a}|\mathbf{s}, \mathbf{z})$, where $\mathbf{z}$ is a latent variable. In one class of methods (Hausman et al., 2018; Gupta et al., 2018), these variables are held constant over the duration of the episode, providing for a time-correlated exploration signal that can enable a single policy to capture multiple skills and rapidly explore plausible behaviors for new tasks by searching in the space of values for $\mathbf{z}$. In another class of methods (Haarnoja et al., 2017, 2018a), the latent variable $\mathbf{z}$ is selected independently at each time step, and the policy $\pi(\mathbf{a}|\mathbf{s}, \mathbf{z})$ has some simple unimodal form (e.g., a Gaussian distribution) conditioned on $\mathbf{z}$, but a complex multimodal form when $\mathbf{z}$ is integrated out. This enables the policy to represent very complex mulitmodal distributions, which can be useful, for example, for capturing the true maximum entropy distribution for an underspecified reward function (e.g., run in all possible directions). It also makes it possible to learn a higher-level policy that uses $\mathbf{z}$ as its action space (Haarnoja et al., 2018a), effectively driving the lower level policy and using it as a distribution over skills. This leads to a natural probabilistic hierarchical reinforcement learning formulation.

## 5.4 Modeling, Intent Inference, and Forecasting

Aside from devising more effective reinforcement learning and optimal control algorithms, maximum entropy reinforcement learning has also been used extensively in the inverse reinforcement learning setting, where the goal is to infer intent, acquire reward functions from data, and predict the behavior of agents (e.g., humans) in the world from observation. Indeed, the use of the term "maximum entropy reinforcement learning" in this article is based on the work of Ziebart and colleagues, who proposed the maximum entropy *inverse* reinforcement learning algorithm (Ziebart et al., 2008) for inferring reward functions and modeling human behavior.

While maximum entropy reinforcement learning corresponds to inference in the graphical model over the variables $\mathbf{s}_t$, $\mathbf{a}_t$, and $\mathcal{O}_t$, *inverse* reinforcement learning corresponds to a learning problem, where the goal is to learn the CPD $p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t)$, given example sequences $\{\mathbf{s}_{1:T,i}, \mathbf{a}_{1:T,i}, \mathcal{O}_{1:T,i}\}$, where $\mathcal{O}_t$ is always true, indicating that the data consists of demonstrations of optimal trajectories. As with all graphical model learning problems, inference takes place in the inner loop of an iterative learning procedure. Exact inference via dynamic programming results in an algorithm where, at each iteration, we solve for the optimal soft value function, compute the corresponding policy, and then use this policy to compute the gradient of the likelihood of the data with respect to the parameters of the CPD $p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t)$. For example, if we use a linear reward representation, such that $p(\mathcal{O}_t|\mathbf{s}_t, \mathbf{a}_t) = \exp(\phi^T f(\mathbf{s}_t, \mathbf{a}_t))$, the learning problem can be expressed as

$$\phi^\star = \arg\max_\phi \sum_i \sum_t \log p(\mathbf{a}_{t,i}|\mathbf{s}_{t,i}, \mathcal{O}_{1:T}, \phi),$$

where computing $\log p(\mathbf{a}_{t,i}|\mathbf{s}_{t,i}, \mathcal{O}_{1:T}, \phi)$ and its gradient requires solving for the optimal policy under the current reward parameters $\phi$.

The same optimism issue discussed in Section 2 occurs in the inverse reinforcement learning setting, where exact inference in the graphical model produces an "optimistic" policy that assumes some degree of control over the system dynamics. For this reason, Ziebart and colleagues proposed the maximum causal entropy framework for inverse reinforcement learning under stochastic dynamics (Ziebart et al., 2010). Although this framework is derived starting from a causal reformulation of the maximum entropy principle, the resulting algorithm is exactly identical to the variational inference algorithm presented in Section 3, and the corresponding learning procedure corresponds to optimizing the variational lower bound with respect to the reward parameters $\phi$.

Subsequent work in inverse reinforcement learning has studied settings where the reward function has a more complex, non-linear representation (Levine et al., 2011; Wulfmeier et al., 2015), and extensions to approximate inference via the Laplace approximation (under known dynamics) (Levine and Koltun, 2012; Dragan et al., 2013) and approximate reinforcement learning (under unknown dynamics) (Finn et al., 2016b; Fu et al., 2018). Aside from inferring reward functions from demonstrations for the purpose of imitation learning, prior work has also sought to leverage the framework of maximum entropy inverse reinforcement learning for inferring the intent of humans, for applications such as robotic assistance (Dragan et al., 2013), brain-computer interfaces (Javdani et al., 2015), and forecasting of human behavior (Huang and Kitani, 2014; Huang et al., 2015).

Recent work has also drawn connections between generative adversarial networks (GANs) (Goodfellow et al., 2014) and maximum entropy inverse reinforcement learning (Finn et al., 2016b,a; Fu et al., 2018; Ho and Ermon, 2016). This connection is quite natural since, just like generative adversarial networks, the graphical model in the maximum entropy reinforcement learning framework is a generative model, in this case of trajectories. GANs avoid the need for explicit estimation of the partition function by noting that, given a model $\hat{p}(\mathbf{x})$ for some true distribution $p(\mathbf{x})$, the optimal classifier for discriminating whether a sample $\mathbf{x}$ came from the model or from the data corresponds to the odds ratio

$$D(\mathbf{x}) = \frac{p(\mathbf{x})}{p(\mathbf{x}) + \hat{p}(\mathbf{x})}.$$

Although $\hat{p}(\mathbf{x})$ is unknown, fitting this "discriminator" and using its gradients with respect to $\mathbf{x}$ to modify $p(\mathbf{x})$ allows for effective training of the generative model. In the inverse reinforcement learning setting, the discriminator takes the form of the reward function. The reward function is learned so as to maximize the reward of the demonstration data and minimize the reward of samples from the current policy, while the policy is updated via the maximum entropy objective to maximize the expectation of the reward and maximize entropy. As discussed by Finn et al., this process corresponds to a generative adversarial network over trajectories, and also corresponds exactly to maximum entropy inverse reinforcement learning (Finn et al., 2016a). A recent extension of this framework also provides for an effective inverse reinforcement learning algorithm in a model-free deep RL context, as well as a mechanism for recovering robust and transferable rewards in ambiguous settings (Fu et al., 2018). A simplification on this setup known as generative adversarial imitation learning (GAIL) (Ho and Ermon, 2016) dispenses with the goal of recovering reward functions, and simply aims to clone the demonstrated policy. In this setup, the algorithm learns the advantage function directly, rather than the reward, which corresponds roughly to an adversarial version of the OptV algorithm (Dvijotham and Todorov, 2010).

A number of prior works have also sought to incorporate probabilistic inference into a model of biological decision making and control (Solway and Botvinick, 2012; Botvinick and An, 2009; Botvinick and Toussaint, 2012; Friston, 2009). The particular frameworks employed in these approaches differ: the formulation proposed by Friston (2009) is similar to the maximum entropy approach outlined in this survey, and also employs the formalism of approximate variational inference. The formulation described by Botvinick and Toussaint (2012) does not use the exponential reward transformation, and corresponds more closely to the "pseudo-likelihood" formulation outlined in Section 5.2.

# 6    Perspectives and Future Directions

In this article, we discussed how the maximization of a reward function in Markov decision process can be formulated as an inference problem in a particular graphical model, and how a set of update equations similar to the well-known value function dynamic programming solution can be recovered as the direct consequence of applying structured variational inference to this graphical model. The classical maximum expected reward formulation emerges as a limiting case of this framework, while the general case corresponds to a maximum entropy variant of reinforcement learning or optimal control, where the optimal policy not only aims to maximize the expected reward, but also aims to maintain high entropy.

The framework of maximum entropy reinforcement learning has already been employed in a range of contexts, as discussed in the previous section, from devising more effective and powerful forward reinforcement learning algorithms, to developing probabilistic algorithms for modeling and reasoning about observed goal-driven behavior. A particularly exciting recent development is the intersection of maximum entropy reinforcement learning and latent variable models, where the graphical model for control as inference is augmented with additional variables for modeling time-correlated stochasticity for exploration (Hausman et al., 2018; Gupta et al., 2018) or higher-level control through learned latent action spaces (Haarnoja et al., 2017, 2018a). The extensibility and compositionality of graphical models can likely be leveraged to produce more sophisticated reinforcement learning methods, and the framework of probabilistic inference can offer a powerful toolkit for deriving effective and convergent learning algorithms for the corresponding models.

Less explored in the recent literature is the connection between maximum entropy reinforcement learning and robust control. Although some work has hinted at this connection (Ziebart, 2010), the potential for maximum entropy reinforcement learning to produce policies that are robust to modeling errors and distributional shift has not been explored in detail. In principle, a policy that is trained to achieve high expected reward under the highest possible amount of injected noise (highest entropy) should be robust to unexpected perturbations at test time. Indeed, recent work in robotics has illustrated that policies trained with maximum entropy reinforcement learning methods (e.g., soft Q-learning) do indeed exhibit a very high degree of robustness (Haarnjoa et al., 2018). However, a detailed theoretical exploration of this phenomenon has so far been lacking, and it is likely that it can be applied more broadly to a range of challenging problems involving domain shift, unexpected perturbations, and model errors.

Finally, the relationship between probabilistic inference and control can shed some light on the design of reward functions and objectives in reinforcement learning. This is an often-neglected topic that has tremendous practical implications: reinforcement learning algorithms typically assume that the reward function is an extrinsic and unchanging signal that is provided as part of the problem definition. However, in practice, the design of the reward function requires considerable care, and the success of a reinforcement learning application is in large part determined by the ability of the user to design a suitable reward function. The control as inference framework suggests a probabilistic interpretation of rewards as log probability of some discrete event variable $\mathcal{O}_t$, and exploring how this interpretation can lead to more interpretable, more effective, and easier to specify reward functions could lead to substantially more practical reinforcement learning methods in the future.

## Acknowledgements

## References

Abdolmaleki, A., Springenberg, J. T., Tassa, Y., Munos, R., Heess, N., and Riedmiller, M. (2018). Maximum a posteriori policy optimisation. In *International Conference on Learning Representations (ICLR)*.

Attias, H. (2003). Planning by probabilistic inference. In *Proceedings of the 9th International Workshop on Artificial Intelligence and Statistics*.

Bagnell, J. A. and Schneider, J. (2003). Covariant policy search. In *International Joint Conference on Artifical Intelligence (IJCAI)*.

Botvinick, M. and An, J. (2009). Goal-directed decision making in prefrontal cortex: a computational framework. In *Advances in Neural Information Processing Systems (NIPS)*.

Botvinick, M. and Toussaint, M. (2012). Planning as inference. *Trends in Cognitive Sciences*, 16(10):485–488.

Dragan, A. D., Lee, K. C. T., and Srinivasa, S. S. (2013). Legibility and predictability of robot motion. In *International Conference on Human-Robot Interaction (HRI)*.

Dvijotham, K. and Todorov, E. (2010). Inverse optimal control with linearly-solvable mdps. In *International Conference on International Conference on Machine Learning (ICML)*.

Finn, C., Christiano, P., Abbeel, P., and Levine, S. (2016a). A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. *CoRR*, abs/1611.03852.

Finn, C., Levine, S., and Abbeel, P. (2016b). Guided cost learning: Deep inverse optimal control via policy optimization. In *International Conference on Machine Learning (ICML)*.

Friston, K. (2009). The free-energy principle: A rough guide to the brain? *Trends in Cognitive Sciences*, 13(7):293–301.

Fu, J., Luo, K., and Levine, S. (2018). Learning robust rewards with adversarial inverse reinforcement learning. In *International Conference on Learning Representations (ICLR)*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Neural Information Processing Systems (NIPS)*.

Gupta, A., Mendonca, R., Liu, Y., Abbeel, P., and Levine, S. (2018). Meta-reinforcement learning of structured exploration strategies. *CoRR*, abs/1802.07245.

Haarnjoa, T., Pong, V., Zhou, A., Dalal, M., Abbeel, P., and Levine, S. (2018). Composable deep reinforcement learning for robotic manipulation. In *International Conference on Robotics and Automation (ICRA)*.

Haarnoja, T., Hartikainen, K., Abbeel, P., and Levine, S. (2018a). Latent space policies for hierarchical reinforcement learning. *CoRR*, abs/1804.02808.

Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. (2017). Reinforcement learning with deep energy-based policies. In *International Conference on Machine Learning (ICML)*.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018b). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *arXiv*.

Hachiya, H., Peters, J., and Sugiyama, M. (2009). Efficient sample reuse in em-based policy search. In *European Conference on Machine Learning (ECML)*.

Hausman, K., Springenberg, J. T., Wang, Z., Heess, N., and Riedmiller, M. (2018). Learning an embedding space for transferable robot skills. In *International Conference on Learning Representations (ICLR)*.

Heess, N., Silver, D., and Teh, Y. W. (2013). Actor-critic reinforcement learning with energy-based policies. In *European Workshop on Reinforcement Learning (EWRL)*.

Ho, J. and Ermon, S. (2016). Generative adversarial imitation learning. In *Neural Information Processing Systems (NIPS)*.

Huang, D., Farahmand, A., Kitani, K. M., and Bagnell, J. A. (2015). Approximate MaxEnt inverse optimal control and its application for mental simulation of human interactions. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Huang, D. and Kitani, K. M. (2014). Action-reaction: Forecasting the dynamics of human interaction. In *European Conference on Computer Vision (ECCV)*.

Javdani, S., Srinivasa, S., and Bagnell, J. A. (2015). Shared autonomy via hindsight optimization. In *Robotics: Science and Systems (RSS)*.

Kaelbling, L. P., Littman, M. L., and Moore, A. P. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285.

Kalman, R. (1960). A new approach to linear filtering and prediction problems. *ASME Transactions journal of basic engineering*, 82(1):35–45.

Kappen, H. J. (2011). Optimal control theory and the linear bellman equation. *Inference and Learning in Dynamic Models*, pages 363–387.

Kappen, H. J., Gómez, V., and Opper, M. (2012). Optimal control as a graphical model inference problem. *Machine Learning*, 87(2):159–182.

Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press.

Levine, S. (2014). *Motor skill learning with local trajectory methods.* PhD thesis, Stanford University.

Levine, S. and Abbeel, P. (2014). Learning neural network policies with guided policy search under unknown dynamics. In *Neural Information Processing Systems (NIPS)*.

Levine, S., Finn, C., Darrell, T., and Abbeel, P. (2016). End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(1).

Levine, S. and Koltun, V. (2012). Continuous inverse optimal control with locally optimal examples. In *International Conference on Machine Learning (ICML)*.

Levine, S. and Koltun, V. (2013a). Guided policy search. In *International Conference on International Conference on Machine Learning (ICML)*.

Levine, S. and Koltun, V. (2013b). Variational policy search via trajectory optimization. In *Advances in Neural Information Processing Systems (NIPS)*.

Levine, S. and Koltun, V. (2014). Learning complex neural network policies with trajectory optimization. In *International Conference on Machine Learning (ICML)*.

Levine, S., Popović, Z., and Koltun, V. (2011). Nonlinear inverse reinforcement learning with gaussian processes. In *Neural Information Processing Systems (NIPS)*.

Minka, T. P. (2001). Expectation propagation for approximate bayesian inference. In *Uncertainty in Artificial Intelligence (UAI)*.

Nachum, O., Norouzi, M., Xu, K., and Schuurmans, D. (2017a). Bridging the gap between value and policy based reinforcement learning. In *arXiv*.

Nachum, O., Norouzi, M., Xu, K., and Schuurmans, D. (2017b). Trust-pcl: An off-policy trust region method for continuous control. *CoRR*, abs/1707.01891.

Neumann, G. (2011). Variational inference for policy search in changing situations. In *International Conference on Machine Learning (ICML)*.

O'Donoghue, B., Munos, R., Kavukcuoglu, K., and Mnih, V. (2017). Pgq: Combining policy gradient and q-learning. In *International Conference on Learning Representations (ICLR)*.

Peters, J., Mülling, K., and Altün, Y. (2010). Relative entropy policy search. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Peters, J. and Schaal, S. (2007). Reinforcement learning by reward-weighted regression for operational space control. In *International Conference on Machine Learning (ICML)*.

Rawlik, K., Toussaint, M., and Vijayakumar, S. (2013). On stochastic optimal control and reinforcement learning by approximate inference. In *Robotics: Science and Systems (RSS)*.

Sallans, B. and Hinton, G. E. (2004). Reinforcement learning with factored states and actions. *Journal of Machine Learning Research*, 5.

Schulman, J., Chen, X., and Abbeel, P. (2017). Equivalence between policy gradients and soft q-learning. In *arXiv*.

Schulman, J., Levine, S., Moritz, P., Jordan, M. I., and Abbeel, P. (2015). Trust region policy optimization. In *International Conference on Machine Learning (ICML)*.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. (2016). High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations (ICLR)*.

Solway, A. and Botvinick, M. (2012). Goal-directed decision making as probabilistic inference: a computational framework and potential neural correlates. *Psychol Rev.*, 119(1):120–154.

Sutton, R. S. (1990). Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *International Conference on Machine Learning (ICML)*.

Theodorou, E. A., Buchli, J., and Schaal, S. (2010). Learning policy improvements with path integrals. In *International Conference on Artificial Intelligence and Statistics (AISTATS 2010)*.

Todorov, E. (2006). Linearly-solvable markov decision problems. In *Advances in Neural Information Processing Systems (NIPS)*.

Todorov, E. (2008). General duality between optimal control and estimation. In *Conference on Decision and Control (CDC)*.

Todorov, E. (2010). Policy gradients in linearly-solvable mdps. In *Neural Information Processing Systems (NIPS)*.

Toussaint, M. (2009). Robot trajectory optimization using approximate inference. In *International Conference on Machine Learning (ICML)*.

Toussaint, M. and Storkey, A. (2006). Probabilistic inference for solving discrete and continuous state markov decision processes. In *International Conference on Machine Learning (ICML)*.

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

Williams, R. J. and Peng, J. (1991). Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3(3):241–268.

Wulfmeier, M., Ondruska, P., and Posner, I. (2015). Maximum entropy deep inverse reinforcement learning. In *Neural Information Processing Systems Conference, Deep Reinforcement Learning Workshop*.

Ziebart, B. (2010). *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. PhD thesis, Carnegie Mellon University.

Ziebart, B. D., Bagnell, J. A., and Dey, A. K. (2010). Modeling interaction via the principle of maximum causal entropy. In *International Conference on Machine Learning (ICML)*.

Ziebart, B. D., Maas, A., Bagnell, J. A., and Dey, A. K. (2008). Maximum entropy inverse reinforcement learning. In *International Conference on Artificial Intelligence (AAAI)*.