Journal of Machine Learning Research 22 (2021) 1-49 Submitted 6/20; Revised 3/21; Published 4/21

Langevin Dynamics for Adaptive Inverse Reinforcement Learning of Stochastic Gradient Algorithms

Vikram Krishnamurthy VIKRAMK@CORNELL.EDU
School of Electrical and Computer Engineering
Cornell University
Ithaca NY, 14853, USA.

George Yin GYIN@UCONN.EDU
Department of Mathematics,
University of Connecticut,
Storrs, CT 06269-1009, USA.

Editor: Andreas Krause

Abstract

Inverse reinforcement learning (IRL) aims to estimate the reward function of optimizing agents by observing their response (estimates or actions). This paper considers IRL when noisy estimates of the gradient of a reward function generated by multiple stochastic gradient agents are observed. We present a generalized Langevin dynamics algorithm to estimate the reward function $R(\theta)$; specifically, the resulting Langevin algorithm asymptotically generates samples from the distribution proportional to $\exp(R(\theta))$. The proposed adaptive IRL algorithms use kernel-based passive learning schemes. We also construct multi-kernel passive Langevin algorithms for IRL which are suitable for high dimensional data and achieve variance reduction. The performance of the proposed IRL algorithms are illustrated on examples in adaptive Bayesian learning, logistic regression (high dimensional problem) and constrained Markov decision processes. We prove weak convergence of the proposed IRL algorithms using martingale averaging methods. We also analyze the tracking performance of the IRL algorithms in non-stationary environments where the utility function $R(\theta)$ has a hyper-parameter that jump changes over time as a slow Markov chain which is not known to the inverse learner. In this case, martingale averaging yields a Markov switched diffusion limit as the asymptotic behavior of the IRL algorithm.

Keywords: passive learning, stochastic gradient algorithm, inverse reinforcement learning, weak convergence, martingale averaging theory, Langevin dynamics, stochastic sampling, inverse Bayesian learning, Constrained Markov Decision process, logistic regression, variance reduction, Bernstein von-Mises theorem, Markov chain hyper-parameter

1. Introduction

Inverse reinforcement learning (IRL) aims to estimate the reward function of optimizing agents by observing their actions (estimates). Classical IRL is off-line: given a data set of actions chosen according to the optimal policy of a Markov decision process, Ng and Russell (2000) formulated a set of inequalities that the reward function must satisfy. In comparison, this paper constructs and analyzes real time IRL algorithms by observing optimizing agents that are performing real time reinforcement learning (RL). The problem we consider is this: Suppose we observe estimates of multiple (randomly initialized) stochastic gradient algorithms (reinforcement learners) that aim to maximize a (possibly non-concave) expected reward. *How to design another stochastic gradient algorithm (inverse learner) to estimate the expected reward function?*

1.1 RL and IRL Algorithms

To discuss the main ideas, we first describe the point of view of multiple agents performing reinforcement learning (RL). These agents act sequentially to perform RL by using stochastic gradient algorithms to maximize a reward function. Let $n = 1, 2 \ldots$ index agents that perform RL sequentially. The sequential protocol is as follows. The agents aim to maximize a possibly non-concave reward $R(\theta) = \mathbb{E}\{r_k(\theta)\}$ where $\theta \in \mathbb{R}^N$. Each agent n runs a stochastic gradient algorithm over the time horizon $k \in \{\tau_n, \tau_n + 1, \ldots, \tau_{n+1} - 1\}$:

$$\begin{gathered}\theta_{k+1} = \theta_k + \varepsilon \, \nabla_\theta r_k(\theta_k), \quad k = \tau_n, \tau_n + 1, \ldots, \tau_{n+1} - 1 \\ \text{initialized independently by } \theta_{\tau_n} \sim \pi(\cdot).\end{gathered} \tag{1}$$

Here $\nabla_\theta r_k(\theta_k)$ denotes the sample path gradient evaluated at θ_k, and τ_n, $n = 1, 2 \ldots$, denote stopping times measurable wrt the σ-algebra generated by $\{\theta_{\tau_n}, \nabla_\theta r_k(\theta_k), k = \tau_n, \tau_n + 1, \ldots\}$. The initial estimate θ_{τ_n} for agent n is sampled independently from probability density function π defined on $\mathbb{R}^N$. Finally, ε is a small positive constant step size.

Next we consider the point of view of an observer that performs *inverse reinforcement learning (IRL)* to estimate the reward function $R(\theta)$. The observer (inverse learner) knows initialization density $\pi(\cdot)$ and only has access to the estimates $\{\theta_k\}$ generated by RL algorithm (1). The observer reconstructs the gradient $\nabla_\theta r_k(\theta_k)$ as $\hat{\nabla}_\theta r_k(\theta_k) = (\theta_{k+1} - \theta_k)/\mu$ for some positive step size μ. The main idea of this paper is to propose and analyze the following IRL algorithm (which is a passive Langevin dynamics algorithm) deployed by the observer:

$$\boxed{\alpha_{k+1} = \alpha_k + \mu\Big[\frac{1}{\Delta^N} K\Big(\frac{\theta_k - \alpha_k}{\Delta}\Big) \frac{\beta}{2} \nabla_\theta r_k(\theta_k) + \nabla_\alpha \pi(\alpha_k)\Big] \pi(\alpha_k) + \sqrt{\mu}\, \pi(\alpha_k)\, w_k, \quad k = 1, 2, \ldots}, \tag{2}$$

initialized by $\alpha_0 \in \mathbb{R}^N$. Here μ and Δ are small positive constant step sizes, $\{w_k, k \geq 0\}$ is an i.i.d. sequence of standard N-variate Gaussian random variables, and $\beta = \varepsilon/\mu$ is a fixed constant. Note that we have expressed (2) in terms of $\nabla_\theta r_k(\theta_k)$ (rather than $\hat{\nabla}_\theta r_\theta(\theta_k)$) since we have absorbed the ratio of step sizes into the scale factor β.

The key construct in (2) is the kernel function[1] $K(\cdot)$. This kernel function is chosen by the observer such that $K(\cdot)$ decreases monotonically to zero as any component of the argument increases to infinity,

$$K(\theta) \geq 0, \quad K(\theta) = K(-\theta), \quad \int_{\mathbb{R}^N} K(\theta) d\theta = 1. \tag{3}$$

An example is to choose the kernel as a multivariate normal $\mathbf{N}(0, \sigma^2 I_N)$ density with $\sigma = \Delta$, i.e.,

$$\frac{1}{\Delta^N} K\Big(\frac{\theta}{\Delta}\Big) = (2\pi)^{-N/2} \Delta^{-N} \exp\Big(-\frac{\|\theta\|^2}{2\Delta^2}\Big),$$

which is essentially like a Dirac delta centered at 0 as $\Delta \to 0$. Our main result stated informally is as follows; see Theorem 1 in Sec.4 for the formal statement.

1. Our use of the term "kernel" stems from non-parametric statistics and passive stochastic approximation. It is not related to reproducing kernels in Hilbert spaces.

Informal Statement of Result. *Based on the estimates* $\{\theta_k\}$ *generated by RL algorithm* (1)*, the IRL algorithm* (2) *asymptotically generates samples* $\{\alpha_k\}$ *from the Gibbs measure*

$$p(\alpha) \propto \exp\bigl(\beta R(\alpha)\bigr), \quad \alpha \in \mathbb{R}^N, \quad \textit{where } \beta = \varepsilon/\mu. \tag{4}$$

To explain the above result, let $\hat{p}$ denote the empirical density function constructed from samples $\{\alpha_k\}$ generated by IRL algorithm (2). Then clearly[2] $\log \hat{p}(\alpha) \propto R(\alpha)$. Thus IRL algorithm (2) serves as a *non-parametric method* for stochastically exploring and reconstructing reward R, given the estimates $\{\theta_k\}$ of RL algorithm (1). Hence based on the estimates $\{\theta_k\}$ generated by RL algorithm (1), IRL algorithm (2) serves as a randomized sampling method for exploring the reward $R(\alpha)$ by simulating random samples from it. Finally, in adaptive Bayesian learning discussed in Sec.3, the RL agents maximize $\log R(\alpha)$ using gradient algorithm (1); then IRL algorithm (2) directly yields samples from $\beta R(\alpha)$.

1.2 Context and Discussion

The stochastic gradient RL algorithm (1) together with non-parametric passive Langevin IRL algorithm (2) constitute our main setup. Figure 1 displays our framework.

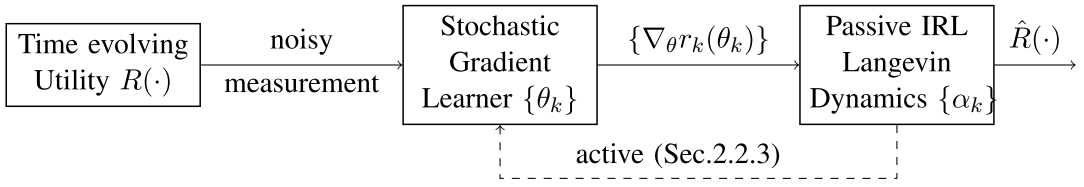

Time evolving Utility $R(\cdot)$
noisy measurement
Stochastic Gradient Learner $\{\theta_k\}$
$\{\nabla_\theta r_k(\theta_k)\}$
Passive IRL Langevin Dynamics $\{\alpha_k\}$
$\hat{R}(\cdot)$
active (Sec.2.2.3)

Figure 1: Schematic of proposed adaptive IRL framework. Multiple agents (learners) compute noisy gradient estimates $\nabla_\theta r_k(\theta_k)$ of a possibly time evolving reward $R(\cdot)$. By observing these gradient estimates, the IRL Langevin dynamics algorithm (2) generates samples $\alpha_k \sim \exp(\beta R(\alpha))$. So reward $R(\cdot)$ can be estimated from the log of the empirical distribution of $\{\alpha_k\}$. The IRL algorithm (2) is *passive*: its estimate α_k plays no role in determining the point θ_k where the learner evaluates gradients $\nabla_\theta r_k(\theta_k)$. Sec.2.2 presents several additional IRL algorithms including a variance reduction algorithm and a non-reversible diffusion. Sec.2.2.3 presents an active IRL where the IRL requests the learner to provide a gradient at α_k, but the learner provides the noisy mis-specified gradient estimate $\nabla_\theta r_k(\alpha_k + v_k)$ where $\{v_k\}$ is a noise sequence. Finally, Sec.5 analyzes the tracking properties of the IRL algorithm when the reward $R(\cdot)$ evolves in time according to an unknown Markov chain.

More abstractly, the IRL problem we address is this: given a sequence of noisy sample path gradients $\{\nabla_\theta r_k(\theta_k)\}$, how to estimate the expected reward $R(\cdot)$? The IRL algorithm (2) builds on classical stochastic gradient algorithms in 3 steps. First, it is *passive*: it does not specify where the RL agents compute gradient estimates. The gradient estimates are evaluated by RL agents at points θ_k, whereas the IRL algorithm requires gradients at α_k. To incorporate these mis-specified gradients, the passive algorithm uses the kernel $K(\cdot)$. Second, a classical passive stochastic gradient algorithm only estimates a local maximum of $R(\cdot)$; in comparison we are interested in non-parametric

2. Since the IRL algorithm does not know the step size ε of the RL, it can only estimate $R(\cdot)$ up to a proportionality constant β. In classical Langevin dynamics β denotes an inverse temperature parameter.

reconstruction (estimation) of the entire reward $R(\cdot)$. Therefore, we use a passive *Langevin dynamics* based algorithm. Finally, we are interested in *tracking* (estimating) time evolving reward functions $R(\cdot)$. Therefore we use a *constant step size*, passive Langevin dynamics IRL algorithm; see point (vii) below.

To give additional insight we now discuss the context, useful generalizations of IRL algorithm (2), and related works in the literature.

(i) *Multiple agents.* The multiple agent RL algorithm (1) is natural in non-convex stochastic optimization problems. Starting from various randomly chosen initial conditions $\theta_{\tau_n} \sim \pi(\cdot)$, the agents evaluate the gradients $\nabla_\theta r_k(\theta_k)$ at various points θ_k to estimate the global maximizer. Since the initializations $\{\theta_{\tau_n}\}$ is a sequence of independent random variables, the RL agents can also act in parallel (instead of sequentially). Given this sequence of gradients $\{\nabla_\theta r_k(\theta_k)\}$, the aim of this paper is to construct IRL algorithms to estimate $R(\theta)$. As an example, motivated by stochastic control involving information theoretic measures (Guan et al., 2014) detailed in Sec.3.1, suppose multiple RL agents run stochastic gradient algorithms to estimate the minimum of a non-convex Kullback Leibler (KL) divergence. By observing these gradient estimates, the passive IRL algorithm (2) reconstructs the KL divergence.[3]

(ii) *Passive IRL.* The IRL algorithm (2) is a Langevin dynamics based gradient algorithm with injected noise $\{w_k\}$. It is a *passive* learning algorithm since the gradients are not evaluated at α_k by the inverse learner; instead the gradients are evaluated at the random points θ_k chosen by the RL algorithm. This passive framework is natural in an IRL. The inverse learner passively observes the RL algorithm and aims to estimate its utility.

We emphasize that the passive Langevin IRL algorithm (1) estimates the utility function $R(\theta)$; see (4). This is unlike classical passive stochastic gradient algorithms that estimate a local stationary point of the utility. To the best of our knowledge, such passive Langevin dynamics algorithms have not been proposed or analyzed - yet such algorithms arise naturally in estimating the utility by observing the estimates from a stochastic gradient algorithm.

The kernel $K(\cdot)$ in (2) effectively weights the usefulness of the gradient $\nabla_\theta r_k(\theta_k)$ compared to the required gradient $\nabla_\alpha r_k(\alpha_k)$. If θ_k and α_k are far apart, then kernel $K((\theta_k - \alpha_k)/\Delta)$ will be small. Then only a small proportion of the gradient estimate $\nabla_\theta r_k(\theta_k)$ is added to the IRL iteration. On the other hand, if $\alpha_k = \theta_k$, (2) becomes a standard Langevin dynamics type algorithm. We refer to Révész (1977); Hardle and Nixdorf (1987); Nazin et al. (1989); Yin and Yin (1996) for the analysis of passive stochastic gradient algorithms. The key difference compared to these works is that we are dealing with a passive Langevin dynamics algorithm, i.e., there is an extra injected noise term involving w_k.

(iii) *Intuition behind passive Langevin IRL algorithm (2).* To discuss the intuition behind (2), we first discuss the classical Langevin dynamics and also a more general reversible diffusion. The

3. To give additional context, multi-agent systems for *deterministic* optimization are studied in Nedic and Ozdaglar (2009) where the aim is to optimize cooperatively the sum of convex objectives corresponding to multiple agents. Agents (represented by nodes in a graph) deploy deterministic sub-gradient algorithms and exchange information to achieve consensus. We consider a *stochastic* optimization framework where agents compute noisy gradient estimates. Our passive IRL uses these gradient estimates to reconstruct (explore) the reward function and uses a (stochastic gradient)) Langevin dynamics algorithm.

classical Langevin dynamics algorithm with fixed step size $\mu > 0$ and deterministic reward $R(\theta)$ is of the form

$$\theta_{k+1} = \theta_k + \mu \, \nabla R(\theta_k) + \sqrt{\mu} \sqrt{\frac{2}{\beta}} \, w_k, \quad k = 1, 2, \dots \tag{5}$$

Indeed (5) is the Euler-Maruyama time discretization of the continuous time diffusion process

$$d\theta(t) = \nabla_\theta R(\theta) + \sqrt{\frac{2}{\beta}} \, dW(t) \tag{6}$$

which has stationary measure $p(\theta)$ given by (4). More generally, assuming $\sigma(\cdot)$ is differentiable, Stramer and Tweedie (1999) studied reversible diffusions of the form

$$d\theta(t) = \left[\frac{\beta}{2} \sigma(\theta) \, \nabla_\theta R(\theta) \, dt + \nabla_\theta \sigma(\theta) \, dt + dW(t)\right] \sigma(\theta), \tag{7}$$

whose Euler-Maruyama time discretization yields

$$\theta_{k+1} = \theta_k + \mu \big[\frac{\beta}{2} \, \nabla_\theta R(\theta_k) + \nabla_\theta \sigma(\theta_k)\big] \, \sigma(\theta_k) + \sqrt{\mu} \, \sigma(\theta_k) \, w_k, \quad k = 1, 2, \dots \tag{8}$$

It is easily verified that reversible diffusion (7) has the same Gibbs stationary measure $p(\theta)$ in (4).

The IRL algorithm (2) substantially generalizes (8) in three ways: First, the gradient is at a mis-specified point θ_k compared to α_k; hence we use the kernel K as discussed in point (ii) above. Second, unlike (8) which uses $\nabla_\theta R(\theta)$, IRL algorithm (2) only has the (noisy) gradient estimate $\nabla_\theta r_k(\theta)$. Finally, we choose $\sigma(\theta)$ as $\pi(\theta)$, namely the initialization density specified in (1), to ensure that the stationary measure is as specified in (4) as explained at the end of Sec.2.1.

The intuition behind the weak convergence of the passive Langevin IRL algorithm (2) is explained in Sec.2.1. It is shown there via stochastic averaging arguments as the kernel converges to a Dirac-delta, the IRL algorithm (2) converges to the reversible diffusion process (7) with stationary measure given by (4).

(iv) *IRL for Markov Decision Process.* Several types of RL based policy gradient algorithms in the Markov decision process (MDP) literature (Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998) fit our framework. As a motivation, we now briefly discuss IRL for an infinite horizon average cost[4] MDP; details are discussed in Sec.3.3. Let $\{x_n\}$ denote a finite state Markov chain with controlled transition probabilities $P_{ij}(u) = \mathbb{P}[x_{n+1} = j | x_n = i, u_n = u]$ where action u_n is chosen from policy $\mathbf{u}_\theta$ parametrized by θ as $u_n = \mathbf{u}_\theta(x_n)$. Solving an average cost MDP (assuming it is unichain (Puterman, 1994)) involves computing the optimal parameter $\theta^* = \sup\{\theta : R(\theta)\}$ where the cumulative reward is

$$R(\theta) = \lim_{T \to \infty} \inf \frac{1}{T} \mathbb{E}_\theta \Big[\sum_{n=1}^{T} \rho(x_n, u_n) \mid x_0 = x\Big], u_n = \mathbf{u}_\theta(x_n) \tag{9}$$

Suppose now that a forward learner runs a policy gradient RL algorithm that evaluates estimates $\nabla_\theta r_k(\theta)$ of $\nabla_\theta R(\theta)$ in order to estimate θ^*. Given these gradient estimates, how can an IRL algorithm estimate $R(\theta)$?

4. As mentioned in Sec.3.3, our IRL algorithms also apply to the simpler discounted cost MDP case.

In Sec.3.3, motivated by widely used fairness constraints in wireless communications, we consider more general average cost *constrained* MDPs (CMDPs), see Altman (1999); Ngo and Krishnamurthy (2010); Borkar and Jain (2010). In CMDPs (Altman, 1999), the optimal policy is randomized. Since the optimal policy is randomized, classical stochastic dynamic programming or Q-learning cannot be used to solve CMDPs as they yield deterministic policies. One can construct a Lagrangian dynamic programming formulation (Altman, 1999) and Lagrangian Q-learning algorithms (Djonin and Krishnamurthy, 2007). Sec.3.3 considers the case where the RL agents deploy a policy gradient algorithm. By observing these gradient estimates, our IRL algorithm reconstructs the Lagrangian of the CMDP.

Notice that our non-parametric setup is different to classical IRL in Ng and Russell (2000) where the inverse learner has access to actions from the optimal policy, knows the controlled transition probabilities, and formulates a set of linear inequalities that the reward function $\rho(x,u)$ satisfies. Our IRL framework only has access to gradient estimates $\nabla_\theta r_k(\theta)$ evaluated at random points θ, and does not require knowledge of the parameters of the CMDP. Also our IRL framework is adaptive (see point (vii) below): the IRL algorithm (2) can track a time evolving $R(\theta)$ due to the transition probabilities or rewards of the MDP evolving over time (and unknown to the inverse learner).

(v) *Multi-kernel IRL.* IRL algorithm (2) requires the gradient $\nabla_\theta r(\theta_k)$ and knowing the density $\pi(\cdot)$. In Sec.2.2.2 we will discuss a two-time scale multi-kernel IRL algorithm, namely (20), that does not require knowledge of the density $\pi(\cdot)$. All that is required is a sequence of samples $\{\nabla_\theta r_k(\theta_i), i = 1 \ldots, L\}$ when the IRL estimate is α_k. The multi-kernel IRL algorithm (20) incorporates variance reduction and is suitable for high dimensional inference. In Sec.2.2, we also discuss several other variations of IRL algorithm (2) including a mis-specified active IRL algorithm where the gradient is evaluated at a point θ_k that is a corrupted value of α_k.

(vi) *Global Optimization vs IRL.* Langevin dynamics based gradient algorithms have been studied as a means for achieving global minimization for non-convex stochastic optimization problems, see for example Gelfand and Mitter (1991). The papers Teh et al. (2016); Raginsky et al. (2017) give a comprehensive study of convergence of the Langevin dynamics stochastic gradient algorithm in a non-asymptotic setting. Also Welling and Teh (2011) studies Bayesian learning, namely, sampling from the posterior using stochastic gradient Langevin dynamics.

Langevin dynamics for global optimization considers the limit as $\beta \to \infty$. In comparison, the IRL algorithms in this paper consider the case of fixed $\beta = \varepsilon/\mu$, since we are interested in sampling from the reward $R(\cdot)$. Also, we consider passive Langevin dynamics algorithms in the context of IRL. Thus the IRL algorithm (2) is non-standard in two ways. First, as mentioned above, it has a kernel to facilitate passive learning. Second, the IRL algorithm (2) incorporates the initialization probability $\pi(\cdot)$ which appears in the RL algorithm (1). Thus (2) is a non-standard generalized Langevin dynamics algorithm (which still has reversible diffusion dynamics).

(vii) *Constant step size Adaptive IRL for Time Evolving Utility.* An important feature of the IRL algorithm (2) is the constant step size μ (as opposed to a decreasing step size). This facilities estimating (adaptively tracking) rewards that evolve over time. Sec.5 gives a formal weak convergence analysis of the asymptotic tracking capability of the IRL algorithm (2) when the reward $R(\cdot)$ jump changes over time according to an unknown Markov chain. The Markov chain constitutes a hyper-parameter in the sense that it is not known or used by the IRL algorithm; it is used in our convergence analysis to determine how well the IRL algorithm can learn a time evolving reward.

The analysis is very different to classical tracking analysis of stochastic gradient algorithms (Benveniste et al., 1990) where the underlying hyper-parameter evolves continuously over time.

The assumptions used in analyzing the adaptation of the IRL algorithm in Sec.5 are similar to those for the passive IRL in Sec.4; the main additional assumption is that the Markov chain's transition matrix is parametrized by a small parameter η. Depending on how η compares to the IRL algorithm step size μ, we analyze three cases of adaptive IRL in Sec.5: (i) the reward jump changes on a slower time scale than the dynamics of the Langevin IRL algorithm, i.e., $\eta = o(\mu)$; (ii) the reward jump changes on the same time scale as the Langevin IRL algorithm, i.e., $\eta = O(\mu)$; (iii) The reward jump changes on a faster time scale compared to the Langevin IRL algorithm, i.e., $\mu = o(\eta)$.

The most interesting (and difficult) case considered in Sec.5 is when the reward changes at the same rate as the IRL algorithm, i.e., $\eta = O(\mu)$. Then stochastic averaging theory yields a Markov switched diffusion limit as the asymptotic behavior of the IRL algorithm. This is in stark contrast to classical averaging theory of stochastic gradient algorithms which yields a deterministic ordinary differential equation (Kushner and Yin, 2003; Benveniste et al., 1990). Due to the constant step size, the appropriate notion of convergence is weak convergence (Kushner and Yin, 2003; Ethier and Kurtz, 1986; Billingsley, 1999). The Markovian hyper-parameter tracking analysis generalizes our earlier work Yin et al. (2004, 2009) in stochastic gradient algorithms to the current case of passive Langevin dynamics with a kernel.

(viii) *Estimating utility functions*. Estimating a utility function given the response of agents is studied under the area of revealed preferences in microeconomics. Afriat's theorem (Afriat, 1967; Diewert, 2012; Varian, 2012) in revealed preferences uses the response of a linearly constrained optimizing agent to construct a set of linear inequalities that are necessary and sufficient for an agent to be an utility maximizer; and gives a set valued estimate of the class of utility functions that rationalize the agents behavior. Different to revealed preferences, the current paper uses noisy gradients to recover the utility function and that too in real time via a constant step size Langevin diffusion algorithm.

We already mentioned classical IRL (Ng and Russell, 2000; Abbeel and Ng, 2004) which aims to estimate an unknown deterministic reward function of an agent by observing the optimal actions of the agent in a Markov decision process setting. Ziebart et al. (2008) uses the principle of maximum entropy for achieving IRL of optimal agents. More abstractly, IRL falls under the area of *imitation learning* (Ho and Ermon, 2016; Osa et al., 2018) which is the process of learning from demonstration. Our IRL approach can be considered as imitation learning from mis-specified noisy gradients evaluated at random points in Euclidean space. We perform *adaptive* (real time) IRL: given samples from a stochastic gradient algorithm (possibly a forward RL algorithm), we propose a Langevin dynamics algorithm to estimate the utility function. Our real-time IRL algorithm facilitates adaptive IRL, i.e., estimating (tracking) time evolving utility functions. In Sec.5, we analyze the tracking properties of such non-stationary IRL algorithms when the utility function jump changes according to an unknown Markov process.

(ix) *Interpretation as a numerical integration algorithm*. Finally, it is helpful to view IRL algorithm (2) as a numerical integration algorithm when the integrand (gradients to be integrated) are presented at random points and the integrand terms are corrupted by noise (noisy gradients). One possible offline approach is to discretize $\mathbb{R}^N$ and numerically build up an estimate of the integral at the discretized points by rounding off the evaluated integrands terms to the nearest discretized point. However, such an approach suffers from the curse of dimensionality: one needs $O(2^N)$ points to construct the integral with a specified level of tolerance. In comparison, the passive IRL algorithm (2) provides a principled real time approach for generating samples from the integral, as depicted by main result (4).

(x) Although our main motivation for *passive* Langevin dynamics stems from IRL, namely estimating a utility function, we mention the interesting recent paper by Kamalaruban et al. (2020) which shows that classical Langevin dynamics yields more robust RL algorithms compared to classic stochastic gradient.[5] In analogy to Kamalaruban et al. (2020), in future work it is worthwhile exploring if our passive Langevin dynamics algorithm can be viewed as a robust version of classical passive stochastic gradient algorithms.

(xi) Finally, we assumed in (1) that the RL agents use a fixed step size ε which is not necessarily known to the inverse learner deploying (2). More generally[5], the step size of each RL agent n in (1) can be chosen as $\varepsilon_n = \varepsilon(1 + \nu_n)$ where $\{\nu_n\}$ is an iid bounded zero mean process. This models the case where the RL agents deploy different step sizes, for example, due to separate hyper-parameter tuning methods, that are not known to the inverse learner. Then our main result (4) continues to hold. In particular, denote $\beta_n = \varepsilon_n/\mu$. Then as explained in Footnote 6 in Sec.2.1, by averaging theory arguments, β_n "averages out"' to $\beta = \varepsilon/\mu$ on the IRL algorithm time scale.

1.3 Organization

The rest of the paper is organized as follows:

1. Sec.2 discusses the IRL algorithm (2), related works in the literature and gives an informal proof of convergence based on averaging theory arguments. Also the following IRL algorithms are discussed:
   (a) A two time scale multi-kernel IRL algorithm with variance reduction. This IRL algorithm is illustrated in a high dimensional example.
   (b) An active IRL algorithm with mis-specified gradient. That is, given the current estimate α_k, the IRL is given a gradient estimate at $\nabla_\theta r_k(\alpha_k + v_k)$ where v_k is a noise process, and the mis-specified point $\alpha_k + v_k$ is known to the IRL algorithm.
   (c) A non-reversible diffusion IRL where a skew symmetric matrix yields a larger spectral gap and therefore faster convergence to the stationary distribution (at the expense of increased computational cost).
2. Sec.3 gives three classes of numerical examples that illustrate our proposed IRL algorithms:
   (a) Learning the KL divergence given noisy gradients. Also IRL for Adaptive Bayesian learning is discussed.
   (b) IRL on a logistic regression classifier involving the adult `a9a` dataset; this is a large dimensional example with $N = 124$ and requires careful use of the proposed multi-kernel IRL algorithm.
   (c) IRL for reconstructing the cumulative reward of an finite horizon constrained Markov decision process (CMDP). Such CMDPs are non-convex in the action probabilities and

5. We thank an anonymous reviewer for bringing this paper to our attention. We also thank the reviewer for suggesting the idea of agent specific step sizes ε_n in point (xi).

have optimal polices that are randomized. We demonstrate how the Langevin-based IRL can learn a from a policy gradient RL algorithm.

The numerical examples are provided as proof of concept rather than a detailed comparison with state-of-the-art.

3. Sec.4 gives a complete weak convergence proof of IRL algorithm (2) using martingale averaging methods. Sec.6 gives a formal proof of convergence of the multi-kernel algorithm (20).
4. Sec.5 provides a formal weak convergence analysis of the asymptotic tracking capability of the IRL algorithm (2) when the utility function jump changes according to a slow (but unknown) Markov chain.
5. Finally, the appendix gives Matlab source codes for the three numerical examples presented in the paper. So the numerical results of this paper are fully reproducible.

2. Informal Proof and Alternative IRL Algorithms

The RL algorithm (1) together with IRL algorithm (2) constitute our main setup. In this section, we first start with an informal proof of convergence of (2) based on stochastic averaging theory; the formal proof is in Sec.4. The informal proof provided below is useful because it gives additional insight into the design of related IRL algorithms. We then discuss several related IRL algorithms including a novel multi-kernel version with variance reduction.

2.1 Informal Proof of Main Result (4)

Since the IRL algorithm (2) uses a constant step size, the appropriate notion of convergence is weak convergence. Weak convergence (for example, (Ethier and Kurtz, 1986)) is a function space generalization of convergence in distribution; function space because we prove convergence of the entire trajectory (stochastic process) rather than simply the estimate at a fixed time (random variable).

A few words about our proof approach. Until the mid 1970s, convergence proofs of stochastic gradient algorithms assumed martingale difference type of uncorrelated noises. The so-called ordinary differential equation (ODE) approach was proposed by Ljung (1977) for correlated noises and decreasing step size, yielding with probability one convergence. This was subsequently generalized by Kushner and coworkers (see for example Kushner (1984)) to weak convergence analysis of constant step size algorithms. The assumptions required in this paper are weaker and the results more general than that used in classical mean square error analysis because we are dealing with suitably scaled sequences of the iterates that are treated as stochastic processes rather than random variables. Our approach captures the dynamic evolution of the algorithm. As a consequence, using weak convergence methods we can also analyze the tracking properties of the IRL algorithms when the parameters are time varying (see Sec.5).

As is typically done in weak convergence analysis, we first represent the sequence of estimates $\{\alpha_k\}$ generated by the IRL algorithm as a continuous-time random process. This is done by constructing the continuous-time trajectory via piecewise constant interpolation as follows: For $t \in [0, T]$, define the continuous-time piecewise constant interpolated processes parametrized by the step size μ as

$$\alpha^{\mu}(t) = \alpha_k, \quad \text{for } t \in [\mu k, \mu k + \mu). \tag{10}$$

Sec.4 gives the detailed weak convergence proof using the martingale problem formulation of Stroock and Varadhan (Ethier and Kurtz, 1986).

Our informal proof of the main result (4) proceeds in two steps:

Step I. We first *fix* the kernel step size Δ and apply stochastic averaging theory arguments: this says that at the slow time scale, we can replace the fast variables by their expected value. For small step sizes ε and $\mu = \varepsilon/\beta$, there are three time scales in IRL algorithm (2):

1. $\{\theta_k\}$ evolves slowly on intervals $k \in \{\tau_n, \tau_{n+1} - 1\}$, and $\{\alpha_k\}$ evolves slowly versus k.
2. We assume that the run-time of the RL algorithm (1) for each agent n is bounded by some finite constant, i.e., $\tau_{n+1} - \tau_n < M$ for some constant M. So $\{\theta_{\tau_n}\} \sim \pi$ is a fast variable compared to $\{\alpha_k\}$.
3. Finally the noisy gradient process $\{\nabla_\theta r_k(\cdot)\}$ evolves at each time k and is a faster variable than $\{\theta_{\tau_n}\}$ which is updated at stopping times τ_n.

With the above time scale separation, there are two levels of averaging involved. First averaging the noisy gradient $\nabla_\theta r_k(\theta_k)$ yields $\nabla_\theta R(\theta)$. Next[6] averaging $\{\theta_{\tau_n}\}$ yields $\theta \sim \pi$. Thus applying averaging theory to IRL algorithm (2) yields the following averaged system:

$$\begin{aligned}\bar\alpha_{k+1} &= \bar\alpha_k + \mu\, \mathbb{E}_{\theta\sim\pi}\Big[\frac{1}{\Delta^N} K\big(\frac{\theta-\bar\alpha_k}{\Delta}\big)\, \frac{\beta}{2}\, \nabla_\theta R(\theta) + \nabla_\alpha \pi(\bar\alpha_k)\Big]\, \pi(\bar\alpha_k) + \sqrt{\mu}\, \pi(\bar\alpha_k)\, w_k \\ &= \bar\alpha_k + \mu \int_{\mathbb{R}^N} \frac{1}{\Delta^N} K\big(\frac{\theta-\bar\alpha_k}{\Delta}\big)\, \frac{\beta}{2}\, \pi(\bar\alpha_k)\nabla_\theta R(\theta)\pi(\theta)d\theta + \pi(\bar\alpha_k)\, \nabla_\alpha\pi(\bar\alpha_k) + \sqrt{\mu}\, \pi(\bar\alpha_k)\, w_k.\end{aligned} \tag{11}$$

Given the sequence $\{\bar\alpha_k\}$, define the interpolated continuous time process $\bar\alpha^\mu$ as in (10). Then as μ goes to zero, $\bar\alpha^\mu$ converges weakly to the solution of the stochastic differential equation

$$\begin{aligned} d\alpha(t) &= \int_{\mathbb{R}^N} \frac{1}{\Delta^N} K\big(\frac{\theta-\alpha}{\Delta}\big) \Big[\frac{\beta}{2}\pi(\alpha)\, \nabla_\theta R(\theta)\, dt\Big]\, \pi(\theta)\, d\theta + \pi(\alpha)\, \nabla_\alpha \pi(\alpha)\, dt + \pi(\alpha)\, dW(t), \\ \alpha(0) &= \alpha_0, \end{aligned} \tag{12}$$

where $W(t)$ is standard Brownian motion. Put differently, the Euler-Maruyama time discretization of (12) yields (11). To summarize (12) is the continuous-time averaged dynamics of IRL algorithm (2). This is formalized in Sec.4.

Step II. Next, we set the kernel step size $\Delta \to 0$. Then $K(\cdot)$ mimics a Dirac delta function and so the asymptotic dynamics of (12) become the diffusion

$$d\alpha(t) = \Big[\frac{\beta}{2}\pi(\alpha)\, \nabla_\alpha R(\alpha)\, dt + \nabla_\alpha \pi(\alpha)\, dt + dW(t)\Big]\, \pi(\alpha), \quad \alpha(0) = \alpha_0 \tag{13}$$

Finally, (13) is a reversible diffusion and its stationary measure is the Gibbs measure $p(\alpha)$ defined in (4). Showing this is straightforward:[7] Recall (Karatzas and Shreve, 1991) that for a generic diffusion process denoted as $dx(t) = f(x)dt + \sigma(x)dW(t)$, the stationary distribution p satisfies

$$\mathcal{L}^* p = \frac{1}{2}\, \mathrm{Tr}[\nabla^2(\Sigma p)] - \mathrm{div}(fp) = 0, \qquad \text{where } \Sigma = \sigma\sigma' \tag{14}$$

6. Recall item (xi) of Sec.1.2 discussed the case where each agent n chooses step size $\varepsilon_n = \varepsilon(1+\nu_n)$. Then $\beta_n = \varepsilon_n/\mu$ is averaged at this time scale yielding β.
7. Note Stramer and Tweedie (1999, Eq.34) has a typographic error in specifying the determinant.

and $\mathcal{L}^*$ is the forward operator. From (13), $f(\alpha) = [\frac{\beta}{2}\pi(\alpha)\,\nabla_\alpha R(\alpha) + \nabla_\alpha \pi(\alpha)]\pi(\alpha)$, $\sigma = \pi(\alpha) I$. Then it is verified by elementary calculus that $p(\alpha) \propto \exp\bigl(\beta R(\alpha)\bigr)$ satisfies (14).

To summarize, we have shown informally that IRL algorithm (2) generates samples from (4). Sec.4 gives the formal weak convergence proof.

(v) *Why not use classical Langevin dynamics?* The passive version of the classical Langevin dynamics algorithm reads:

$$\alpha_{k+1} = \alpha_k + \mu\,\frac{1}{\Delta^N}\,K\bigl(\frac{\theta_k - \alpha_k}{\Delta}\bigr)\,\nabla r_k(\theta_k) + \sqrt{\mu}\,\sqrt{\frac{2}{\beta}}\,w_k, \quad k = 1, 2, \ldots \tag{15}$$

where θ_k are computed by RL (1). Then averaging theory (as $\mu \to 0$ and then $\Delta \to 0$) yields the following asymptotic dynamics (where $W(t)$ denotes standard Brownian motion)

$$d\alpha(t) = \nabla_\alpha R(\alpha)\,\pi(\alpha) dt + \sqrt{\frac{2}{\beta}} dW(t), \quad \alpha(0) = \alpha_0 \tag{16}$$

Then the stationary distribution of (16) is proportional to $\exp(\beta \int [\nabla_\alpha R(\alpha)\,\pi(\alpha)] d\alpha)$. Unfortunately, this is difficult to relate to $R(\alpha)$ and therefore less useful. In comparison, the generalized Langevin algorithm (2) yields samples from stationary distribution proportional to $\exp(\beta R(\alpha))$ from which $R(\alpha)$ is easily estimated (as discussed below (4)). This is the reason why we will use the passive generalized Langevin dynamics (2) for IRL instead of the passive classical Langevin dynamics (15).

2.2 Alternative IRL Algorithms

IRL algorithm (2) is the vanilla IRL algorithm considered in this paper and its formal proof of convergence is given in Sec.4. In this section we discuss several variations of IRL algorithm (2). The algorithms discussed below include a passive version of the classical Langevin dynamics, a two-time scale multi-kernel MCMC based IRL algorithm (for variance reduction) and finally, a non-reversible diffusion algorithm. The construction of these algorithms are based on the informal proof discussed above.

2.2.1 Passive Langevin Dynamics Algorithms for IRL

IRL algorithm (2) can be viewed as a passive modification of the generalized Langevin dynamics proposed in Stramer and Tweedie (1999). Since generalized Langevin dynamics includes classical Langevin dynamics as a special case, it stands to reason that we can construct a passive version of the classical Langevin dynamics algorithm. Indeed, instead of (2), the following passive Langevin dynamics can be used for IRL (initialized by $\alpha_0 \in \mathbb{R}^N$):

$$\boxed{\alpha_{k+1} = \alpha_k + \mu\,\frac{1}{\Delta^N}\,K\bigl(\frac{\theta_k - \alpha_k}{\Delta}\bigr)\,\frac{\beta}{2\,\pi(\alpha_k)}\,\nabla_\theta r_k(\theta_k) + \sqrt{\mu}\,w_k, \quad k = 1, 2, \ldots} \tag{17}$$

Note that this algorithm is different to (15) due to the term $\pi(\alpha_k)$ in the denominator, which makes a crucial difference. Indeed, unlike (15), algorithm (17) generates samples from (4), as we now explain: By stochastic averaging theory arguments as μ goes to zero, the interpolated processes α^μ converges weakly to (where $W(t)$ below is standard Brownian motion)

$$d\alpha(t) = \int_{\mathbb{R}^N} \frac{1}{\Delta^N}\,K\bigl(\frac{\theta - \alpha}{\Delta}\bigr)\,\left[\frac{\beta}{2\,\pi(\alpha)}\,\nabla_\theta R(\theta)\,dt\right]\pi(\theta)\,d\theta + dW(t), \quad \alpha(0) = \alpha_0 \tag{18}$$

Again as $\Delta \to 0$, $K(\cdot)$ mimics a Dirac delta function and so the $\pi(\cdot)$ in the numerator and denominator cancel out. Therefore the asymptotic dynamics become the reversible diffusion

$$d\alpha(t) = \frac{\beta}{2} \nabla_\alpha R(\alpha)\, dt + dW(t), \quad \alpha(0) = \alpha_0 \tag{19}$$

Note that (19) is the classical Langevin diffusion and has stationary distribution p specified by (4). So algorithm (17) asymptotically generates samples from (4).

Finally, we note that Algorithm (17) can be viewed as a special case of IRL algorithm (2) since its limit dynamics (19) is a special case of the limit dynamics (13) with $\pi(\cdot) = 1$.

2.2.2 VARIANCE REDUCTION FOR HIGH DIMENSIONAL IRL

For large dimensional problems (e.g., $N = 124$ in the numerical example of Sec.3), the passive IRL algorithm (2) can take a very large number of iterations to converge to its stationary distribution. This is because with high probability, the kernel $K(\theta_k, \alpha_k)$ will be close to zero and so updates of α_k will occur very rarely.

There is strong motivation to introduce variance reduction in the algorithm. Below we propose a two time step, multi-kernel variance reduction IRL algorithm motivated by importance sampling. Apart from the ability to deal with high dimensional problems, the algorithm also does not require knowledge of the initialization probability density $\pi(\cdot)$.

Suppose the IRL operates at a slower time scale than the RL algorithm. At each time k (on the slow time scale), by observing the RL algorithm, the IRL obtains a pool of samples of the gradients $\nabla_\theta r_k(\theta_{k,i})$ evaluated at a large number of points $\theta_{k,i}$, $i = 1, 2, \ldots, L$ (here i denotes the fast time scale). As previously, each sample $\theta_{k,i}$ is chosen randomly from $\pi(\cdot)$. Given these sampled derivatives, we propose the following multi-kernel IRL algorithm:

$$\boxed{\alpha_{k+1} = \alpha_k + \mu \frac{\beta}{2} \frac{\sum_{i=1}^L p(\alpha_k|\theta_{k,i}) \nabla_\theta r_k(\theta_{k,i})}{\sum_{l=1}^L p(\alpha_k|\theta_{k,l})} + \sqrt{\mu} w_k, \quad \theta_{k,i} \sim \pi(\cdot)} \tag{20}$$

In (20), we choose the conditional probability density function $p(\theta|\alpha)$ as follows:

$$p(\alpha|\theta) = p_v(\theta - \alpha) \quad \text{where } p_v(\cdot) = \mathbf{N}(0, \sigma^2 I_N). \tag{21}$$

For notational convenience, for each α, denote the normalized weights in (20) as

$$\gamma_{k,i}(\alpha) = \frac{p(\alpha|\theta_{k,i})}{\sum_{l=1}^L p(\alpha|\theta_{k,l})} \quad i = 1, \ldots, L \tag{22}$$

Then these L normalized weights qualify as symmetric kernels in the sense of (3). Thus IRL algorithm (20) can be viewed as a multi-kernel passive stochastic approximation algorithm. Note that the algorithm does not require knowledge of $\pi(\cdot)$.

Since for each k, the samples $\{\theta_{k,i}, i = 1, \ldots, L\}$ are generated i.i.d. random variables, it is well known from self-normalized importance sampling Cappe et al. (2005) that as $L \to \infty$, then for fixed α,

$$\sum_{i=1}^L \gamma_{k,i}(\alpha)\, \nabla_\theta r_k(\theta_{k,i}) \to \mathbb{E}\{\nabla_\theta r_k(\theta)|\alpha\} \quad \text{w.p.1}, \tag{23}$$

provided $\mathbb{E}|p(\theta|\alpha)\,\nabla r_\theta(\theta)| < \infty$. Similar results can also be established more generally if $\{\theta_{k,i}, i = 1,\dots,L\}$ is a geometrically ergodic Markov process with stationary distribution $\pi(\cdot)$.

Remark: Clearly the conditional expectation $\mathbb{E}\{\nabla_\theta r_k(\theta)|\alpha_k\}$ always has smaller variance than $\nabla_\theta r_k(\theta)$; therefore variance reduction is achieved in IRL algorithm (20). In sequential Markov chain Monte Carlo (particle filters), to avoid degeneracy, one resamples from the pool of "particles" $\{\theta_i, i = 1\dots,L\}$ according to the probabilities (normalized weights) γ_i. For large L, the resulting resampled particles have a density $p(\theta|\alpha_k)$. However, we are only interested in computing an estimate of the gradient (and not in propagating particles over time). So we use the estimate $\sum_i \gamma_{k,i}\nabla_\theta r_k(\theta_{k,i})$ in (20); this always has a smaller variance than resampling and then estimating the gradient; see Ross (2013, Sec.12.6) for an elementary proof.

Why not use the popular MCMC tool of sequential importance sampling with resampling? Such a process resamples from the pool of particles and pastes together components of θ_i from other more viable candidates θ_j. As a result, L composite vectors are obtained, which are more viable. However, since our IRL framework is passive, this is of no use since we cannot obtain the gradient for these L composite vectors. Recall that in our passive framework, the IRL has no control over where the gradients $\nabla_\theta r_k(\theta)$ are evaluated.

Informal Analysis of IRL algorithm (20). By stochastic averaging theory arguments as μ goes to zero, the interpolated process α^μ from IRL algorithm (20) converges weakly to

$$d\alpha(t) = \int_{\mathbb{R}^N} \frac{\beta}{2}\,\nabla_\theta R(\theta)\,p\big(\theta|\alpha(t)\big)\,d\theta\,dt + dW(t), \qquad \alpha(0)=\alpha_0 \tag{24}$$

where $W(t)$ is standard Brownian motion. Notice that even though θ_i are sampled from the density $\pi(\cdot)$, the above averaging is w.r.t. the conditional density $p(\theta|\alpha)$ because of (23). For small variance σ^2, by virtue of the classical Bernstein von-Mises theorem (Van der Vaart, 2000), the conditional density $p(\theta|\alpha)$ in (24) acts as a Dirac delta yielding the classical Langevin diffusion

$$d\alpha(t) = \frac{\beta}{2}\,\nabla_\alpha R(\alpha(t))\,dt + dW(t) \tag{25}$$

Therefore algorithm (20) generates samples from distribution (4). The formal proof is in Sec.6.

2.2.3 ACTIVE IRL WITH MIS-SPECIFIED GRADIENT

Thus far we have considered the case where the RL algorithm provides estimates $\nabla_\theta r_k(\theta_k)$ at randomly chosen points independent of the IRL estimate α_k. In other words, the IRL is passive and has no role in determining where the RL algorithm evaluates gradients.

We now consider a modification where the RL algorithm gives a noisy version of the gradient evaluated at a stochastically perturbed value of α_k. That is, when the IRL estimate is α_k, it requests the RL algorithm to provide a gradient estimate $\nabla_\theta r_k(\alpha_k)$. But the RL algorithm evaluates the gradient at a mis specified point $\theta_k = \alpha_k + v_k$, namely, $\nabla_\theta r_k(\alpha_k + v_k)$. Here $v_k \sim \mathbf{N}(0, \sigma^2 I_N)$ is an i.i.d. sequence. The RL algorithm then provides the IRL algorithm with θ_k and $\nabla_\theta r_k(\theta_k)$. So, instead of θ_k being independent of α_k, now θ_k is conditionally dependent on α_k as

$$p(\theta_k|\alpha_k) = \frac{1}{(2\pi)^N\,\sigma^N}\exp(-\frac{1}{2\sigma^2}\|\theta_k-\alpha_k\|^2), \qquad \theta_k,\alpha_k\in\mathbb{R}^N \tag{26}$$

In other words, the IRL now actively specifies where to evaluate the gradient; however, the RL algorithm evaluates a noisy gradient and that too at a stochastically perturbed (mis-specified) point θ_k.

The active IRL algorithm we propose is as follows:

$$\alpha_{k+1} = \alpha_k + \mu \frac{1}{\Delta^N} K\big(\frac{\theta_k - \alpha_k}{\Delta}\big) \frac{\beta}{2\, p(\theta_k|\alpha_k)} \nabla_\theta r_k(\theta_k) + \sqrt{\mu} w_k, \quad \text{where } \theta_k = \alpha_k + v_k \tag{27}$$

The proof of convergence again follows using averaging theory arguments. Since $\{\theta_k\} \sim p(\theta|\alpha_k)$ is the fast signal and $\{\alpha_k\}$ is the slow signal, the averaged system is

$$d\alpha(t) = \int_{\mathbb{R}^N} \frac{1}{\Delta^N} K\big(\frac{\theta - \alpha}{\Delta}\big) \frac{\beta}{2\, p(\theta|\alpha(t))} \nabla_\theta R(\theta)\, p(\theta|\alpha(t))\, d\theta\, dt + dW(t)$$

So the $p(\theta|\alpha(t))$ cancel out in the numerator and denominator. As $\Delta \to 0$, the kernel acts as a Dirac delta thereby yielding the classical Langevin diffusion (25).

Remark: The active IRL algorithm (27) can be viewed as an idealization of the multi-kernel IRL algorithm (20). The multi-kernel algorithm constructs weights to approximate sample from the conditional distribution $p(\theta|\alpha)$. In comparison, the active IRL has direct measurements from this conditional density. So the active IRL can be viewed as an upper bound to the performance of the multi-kernel IRL Another motivation is inertia. Given the dynamics of the RL algorithm, it may not be possible to the RL to abruptly jump to evaluate a gradient at α_k, at best the RL can only evaluate a gradient at a point $\alpha_k + v_k$. A third motivation stems from mis-specification: if the IRL represents a machine (robot) learning from a human, it is difficult to specify to the human exactly what policy α_k to perform. Then $\theta_k = \alpha_k + v_k$ can be viewed as an approximation to this mis-specification.

2.2.4 NON-REVERSIBLE DIFFUSION FOR IRL

So far we have defined four different passive Langevin dynamics algorithms for IRL, namely (2), (17), (20), and (27). These algorithms yield reversible diffusion processes that asymptotically sample from the stationary distribution (4). It is well known (Hwang et al., 1993, 2005; Pavliotis, 2014) that adding a skew symmetric matrix to the gradient always improves the convergence rate of Langevin dynamics to its stationary distribution. That is for any $N \times N$ dimensional skew symmetric matrix $S = -S'$, the non-reversible diffusion process

$$d\alpha(t) = \frac{\beta}{2} (I_N + S) \nabla_\alpha R(\alpha) dt + dW(t), \quad \alpha(0) = \alpha_0 \tag{28}$$

has a larger spectral gap and therefore converges to stationary distribution $\pi(\alpha)$ faster than (13). The resulting IRL algorithm obtained by a Euler-Maruyama time discretization of (28) and then introducing a kernel $K(\cdot)$ is

$$\boxed{\alpha_{k+1} = \alpha_k + \mu \frac{1}{\Delta^N} K\big(\frac{\theta_k - \alpha_k}{\Delta}\big) \frac{\beta\, (I_N + S)}{2\, \pi(\alpha_k)} \nabla_\theta r_k(\theta_k) + \sqrt{\mu}\, w_k, \quad k = 1, 2, \dots} \tag{29}$$

initialized by $\alpha_0 \in \mathbb{R}^N$. Again a stochastic averaging theory argument shows that IRL algorithm (29) converges weakly to the non-reversible diffusion (28). In numerical examples, we found empirically that the convergence of (29) is faster than (2) or (17). However, the faster convergence comes at the expense of an order of magnitude increased computational cost. The computational cost of IRL algorithm (29) is $O(N^2)$ at each iteration due to multiplication with skew symmetric matrix S. In comparison the computational costs of IRL algorithms (2) and (17) are each $O(N)$.

3. Numerical Examples

This section presents three examples to illustrate the performance of the proposed IRL algorithms.

3.1 Example 1. IRL for Bayesian KL divergence and Posterior Reconstruction

This section illustrates the performance of our proposed IRL algorithms in reconstructing the Kullback Leibler (KL) divergence and multi-modal posterior distribution. Our formulation is a stochastic generalization of adaptive Bayesian learning in Welling and Teh (2011) as explained below.

Motivation. Exploring and estimating the KL divergence of a multimodal posterior distribution is important in Bayesian inference (Robert and Casella, 2013), maximum likelihood estimation, and also stochastic control with KL divergence cost (Guan et al., 2014). To motivate the problem, suppose random variable θ has prior probability density $p(\theta)$. Let θ^o denote a fixed (true) value of θ which is unknown to the optimizing agents and inverse learner. Given a sequence of observations $y_{1:T} = (y_1, \dots, y_T)$, generated from distribution $p(y_{1:T}|\theta^o)$, the KL divergence of the posterior distribution is

$$J(\theta^o, \theta) = \mathbb{E}_{\theta^o}\{\log p(\theta^o|y_{1:T}) - \log p(\theta|y_{1:T})\} = \int \log \frac{p(\theta^o|y_{1:T})}{p(\theta|y_{1:T})}\, p(y_{1:T}|\theta^o) dy_{1:T} \tag{30}$$

It is well known (via Jensen's inequality) that the global minimizer θ^* of $J(\theta^o, \theta)$ is θ^o. Therefore minimizing the KL divergence yields a consistent estimator of θ^o. Moreover, under mild stationary conditions, when the prior is non-informative (and so possibly improper), the Shannon-McMillan-Breiman theorem (Barron, 1985) implies that the global minimizer of the KL divergence converges with probability 1 to the maximum likelihood estimate as $T \to \infty$. So there is strong motivation to explore and estimate the KL divergence.

Typically the KL divergence $J(\theta^o, \theta)$ is non-convex in θ. So we are in the non-convex optimization setup of (1) where multiple agents seek to estimate the global minimizer of the KL divergence.

3.1.1 Model Parameters

We consider a stochastic optimization problem where a RL system chooses actions u_k from randomized policy $p(\theta|y_{1:T})$. In order to learn the optimal policy, the RL system aims to estimate the global minimizer $\theta^* = \arg\min_\theta J(\theta^o, \theta)$; see for example Guan et al. (2014) for motivation of KL divergence minimization in stochastic control. Then by observing the gradient estimates of the RL agents, we will use our proposed passive IRL algorithms to reconstruct the KL divergence.

Ignoring the constant term $p(\theta^o|y_{1:T})$ in (30), minimizing $J(\theta^o, \theta)$ wrt θ is equivalent to maximizing the relative entropy $R(\theta) = \mathbb{E}_{\theta^o}\{\log p(\theta|y_{1:T})\}$. So multiple RL agents aim to solve the following non-concave stochastic maximization problem: Find

$$\theta^* = \arg\max_\theta R(\theta), \qquad \text{where } R(\theta) = \mathbb{E}_{\theta^o}\{\log p(\theta|y_{1:T})\} \tag{31}$$

In our numerical example we choose $\theta = [\theta(1), \theta(2)]' \in \mathbb{R}^2$ and θ^o is the true parameter value which is unknown to the learner. The prior is $p(\theta) = \mathbf{N}(0, \Sigma)$ where $\Sigma = \text{diag}[10, 2]$. The observations y_k are independent and generated from the multi-modal mixture likelihood

$$y_k \sim p(y|\theta^o) = \frac{1}{2}\mathbf{N}(\theta^o(1), 2) + \frac{1}{2}\mathbf{N}(\theta^o(1) + \theta^o(2), 2)$$

Since $y_1, \dots, y_T$ are independent and identically distributed, the objective $R(\theta)$ in (31) is

$$R(\theta) = \mathbb{E}_{\theta^o}\{\log p(\theta) + T\,\log p(y|\theta)\} + \text{ constant indpt of } \theta \tag{32}$$

For true parameter value $\theta^o = [0,1]'$, it can be verified that the objective $R(\theta)$ is non-concave and has two maxima at $\theta = [0,1]'$ and $\theta = [1,-1]'$.

3.1.2 Classical Langevin Dynamics

To benchmark the performance of our passive IRL algorithms (discussed below), we ran the classical Langevin dynamics algorithm:

$$\theta_{k+1} = \theta_k + \mu\,\frac{\beta}{2}\nabla_\theta r_k(\theta_k) + \sqrt{\mu}\,w_k, \quad k = 1, 2, \dots, \tag{33}$$

Note that the classical Langevin dynamics (33) evaluates the gradient estimate $\nabla_\theta r_k(\theta_k)$ unlike our passive IRL algorithm which has no control of where the gradient is evaluated. Figure 2 displays both the empirical histogram and a contour plot of the estimate $R(\theta)$ generated by classical Langevin dynamics. The classical Langevin dynamics can be viewed as an upper bound for the performance of our passive IRL algorithm; since our passive algorithm cannot specify where the gradients are evaluated.

3.1.3 Passive IRL Algorithms

We now illustrate the performance of our proposed passive IRL algorithms for the above model. Recall that the framework comprises two parts: First, multiple RL agents run randomly initialized stochastic gradient algorithms to maximize $R(\theta)$. Second, by observing these gradients, our passive IRL Langevin based algorithms construct a non-parametric estimate of the $R(\theta)$. We discuss these two parts below:

1. Multiple agent Stochastic Gradient Algorithm. Suppose multiple RL agents aim to learn the optimal policy by estimating the optimal parameter θ^*. To do so, the agents use the stochastic gradient algorithm (1):

$$\begin{aligned} \theta_{k+1} &= \theta_k + \varepsilon \nabla_\theta r_k(\theta_k) \\ \nabla_\theta r_k(\theta_k) &= \nabla_\theta \log p(\theta_k) + T\,\nabla_\theta \log p(y_k|\theta_k) \end{aligned} \tag{34}$$

with multiple random initializations, depicted by agents $n = 1, 2, \dots$. For each agent n, the initial estimate was sampled randomly as $\theta_{\tau_n} \sim \pi(\cdot) = \mathbf{N}(0, I_{2\times 2})$. Each agent runs the gradient algorithm for 100 iterations with step size $\varepsilon = 10^{-3}$ and the number of agents is 10^5. Thus the sequence $\{\theta_k; k = 1, \dots 10^7\}$ is generated.

2. IRL algorithms and performance. Given the sequence of estimates $\{\theta_k\}$ generated by the RL agents above, and initialization density π, the inverse learner aims to estimate $R(\theta)$ in (32) by generating samples $\{\alpha_k\}$ from $\exp(\beta R(\theta))$. Note that the IRL algorithm has no knowledge of $p(\theta)$ or $p(y|\theta)$. Since the inverse learner has no control of where the reinforcement learner evaluates its gradients, we are in passive IRL setting. We ran the IRL algorithm (2) with kernel $K(\theta,\alpha) \propto \exp(-\frac{\|\alpha-\theta\|^2}{0.02})$, step size $\mu = 5 \times 10^{-4}$, $\beta = 1$. Figure 3 displays both the empirical histogram and a contour plot. Notice that the performance of our IRL is very similar to classical Langevin dynamics (where the gradients are fully specified).

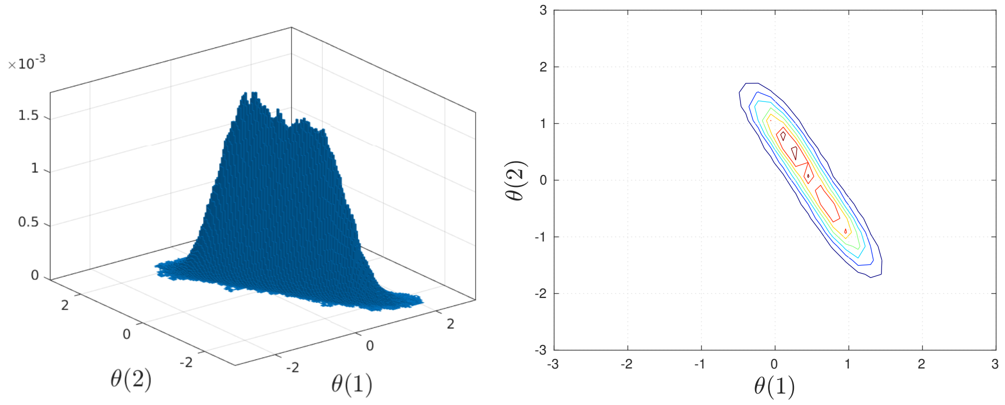


Figure 2: Classical Langevin dynamics (ground truth)

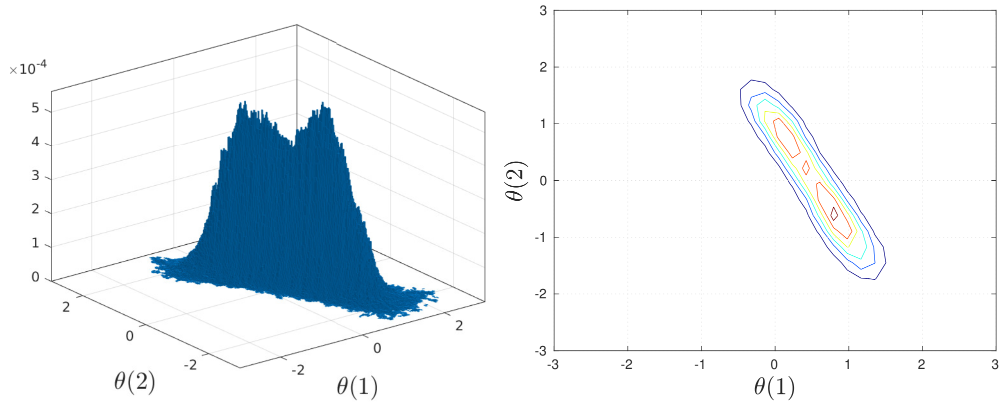


Figure 3: IRL Algorithm (2)

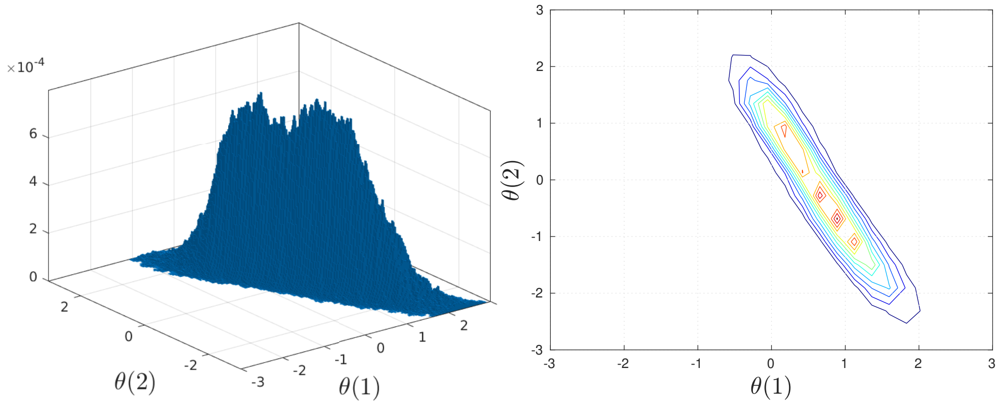


Figure 4: Two time scale multi-kernel IRL Algorithm (20)

We compared the performance of the classical Langevin with the passive Langevin IRL algorithm averaged over 100 independent runs. The comparison is with respect to the variational distance[8] $d(1)$ and $d(2)$ between the two marginals of the empirical density $p(\theta) \propto \exp(R(\theta))$. The values obtained from our simulations are

$$d(1) = 0.0122, \quad d(2) = 0.0202. \tag{35}$$

Finally, we illustrate the performance of the two-time scale multikernel algorithm (20). Recall this algorithm does not require knowledge of the initialization probabilities $\pi(\cdot)$. Figure 4 displays both the empirical histogram and a contour plot. Again the performance of the IRL is very similar to the classical Langevin dynamics performance.

3.1.4 MULTIPLE INVERSE LEARNERS

We also considered the case where multiple inverse learners act in parallel. Suppose each inverse learner $l \in \{1, 2, \dots, L\}$ deploys IRL algorithm (2) with its own noise sample path denoted by $\{w_k^{(l)}\}$, which is independent of other inverse learners. Obviously, if the estimate $\alpha_k^{(l)}$ of one of the inverse learners (say l) is close to θ_k, then $\nabla_\theta r_k(\theta_k)$ is a more accurate gradient estimate for $\nabla_\theta r_k(\alpha_k^{(l)})$. However, for high dimensional problems, our numerical experiments (not presented here) show very little benefit unless the number of inverse learners is chosen as $L = O(2^N)$ which is intractable.

3.1.5 IRL FOR ADAPTIVE BAYESIAN LEARNING

Having discussed reconstructing the KL divergence via IRL, we now extend the Bayesian learning framework proposed in Welling and Teh (2011) to our IRL framework.

Bayesian Learning. First a few words about the Bayesian learning framework in Welling and Teh (2011). In comparison to the stochastic optimization problem (31), they consider a *fixed* sample path $y_{1:T}$ and the associated *deterministic* optimization problem of finding global maximizers of

$$R(\theta) = \log p(\theta | y_{1:T}). \tag{36}$$

Welling and Teh (2011) use the classical Langevin dynamics to generate samples from the posterior $p(\theta|y_{1:T})$ as follows: First, since $y_1, \dots, y_T$ are independent,

$$\nabla_\theta \log p(\theta|y_{1:T}) \propto \nabla_\theta \log p(\theta) + \sum_{k=1}^{T} \nabla_\theta \log p(y_k|\theta) \tag{37}$$

Next it is straightforward to see that T iterations of the classical Langevin algorithm (or a fixed step size deterministic gradient ascent algorithm) using the gradient $\nabla_\theta \log p(\theta) + T\, \nabla_\theta \log p(y_k|\theta)$ is identical to running T sweeps of the algorithm through the sequence $y_{1:T}$ with gradient (37). So Welling and Teh (2011) run the classical Langevin algorithm using the gradient

$$\nabla_\theta \log p(\theta) + T\, \nabla_\theta \log p(y_k|\theta).$$

8. Recall the variational distance is half the L_1 norm

Notice unlike the KL estimation framework (31) which has an expectation $\mathbb{E}_{\theta^o}$ over the observations, the underlying optimization of $\log p(\theta|y_{1:T})$ is *deterministic* since we have a fixed sequence $y_{1:T}$. Then clearly the Langevin dynamics generates samples from the stationary distribution

$$\pi(\theta) = \exp\big(\log(p(\theta|y_{1:T}))\big) = p(\theta|y_{1:T}) \tag{38}$$

namely, the posterior distribution.[9] So the classical Langevin algorithm which sweeps repeatedly through the dataset $y_{1:T}$ generates samples from the posterior distribution - this is the main idea of Bayesian learning in Welling and Teh (2011).

IRL. We now consider IRL in this Bayesian learning framework to reconstruct the posterior density. Given the sample path $y_{1:T}$, suppose multiple forward learners seek to estimate the maximum (mode) of the multimodal posterior $\log p(\theta|y_{1:T})$. The agents run the (deterministic) gradient ascent algorithm (1) with gradient

$$\nabla_\theta r_k(\theta_k) = \nabla_\theta \log p(\theta_k) + T\,\nabla_\theta \log p(y_k|\theta_k)$$

The IRL problem we consider is: By passively observing these gradients, how can the IRL algorithm reconstruct the posterior distribution $p(\theta|y_{1:T})$? We use our IRL algorithm (2). The implementation of IRL algorithm (2) follows the Welling and Teh (2011) setup: The RL agents choose random initializations $\theta_0 \sim \pi$ and then run gradient algorithms sweeping repeatedly through the dataset $y_{1:T}$. The IRL algorithm (2) passively views these estimates $\{\theta_k\}$ and reconstructs the posterior distribution $p(\theta|y_{1:T})$ from these estimates.

We now illustrate the performance of IRL algorithm (2) in this Bayesian learning setup. For the same parameters as in the example above (recall $T = 100$), Table 1 compares the performance of the classical Langevin algorithm and our IRL algorithm with the Metropolis Hastings sampler. The Metropolis Hastings sampler can be considered as the ground truth for the posterior $p(\theta|y_{1:100})$.

	Classical Langevin	Passive IRL Langevin
$d(1)$	0.0213	0.0264
$d(2)$	0.0229	0.0305

Table 1: Variational distance between marginals and Metropolis Hastings algorithm

3.2 Example 2. IRL with Logistic Regression Classifier

We now consider a high dimensional IRL problem ($N = 124$) on the benchmark adult `a9a` dataset. Performing IRL, i.e., generating samples from a 124 dimensional probability density that represents the utility, is challenging and requires use of the multi-kernel variance reduced IRL algorithm (20).

SETUP

In a logistic regression model parameterized by $\theta \in \mathbb{R}^N$, the observations (labels) $y_k \in \{0, 1\}$ are assumed to be generated probabilistically from

$$P(y_k = 1|\theta) = \sigma(\psi_k'\theta) = \frac{1}{1+\exp(-\psi_k'\theta)}, \quad \theta \in \mathbb{R}^N$$

9. This is in contrast to our KL divergence estimation setup (32) where the stationary distribution is $\pi(\theta) = \exp\big(\mathbb{E}_{\theta^o}\{\log p(\theta|y_{1:T})\}\big)$ and $\mathbb{E}_{\theta^o}$ denotes expectation wrt $p(y_{1:T}|\theta^o)$.

Here $\psi_k \in \mathbb{R}^N$ is known input vector at time k and is called the feature.

We consider a Bayesian setting where the prior of θ is assumed to be an N-variate Laplacian density with independent components. So the prior is

$$p(\theta) \propto \exp(-\sum_{i=1}^{N} |\theta(i)|).$$

As in the Bayesian learning setup (36) above, given the fixed sequence $y_{1:T}$, the RL agents aim to find the global maximizer of

$$R(\theta) = \log p(\theta | y_{1:T}) \tag{39}$$

To do so, the RL agents use the gradient algorithm

$$\theta_{k+1} = \theta_k + \varepsilon \big[\nabla_\theta \log p(\theta_k) + T \, \nabla_\theta \log p(y_k | \theta_k) \big]. \tag{40}$$

with multiple sweeps over the dataset. Note that for the logistic model, $\nabla_\theta \log p(\theta_k) = -\operatorname{sgn}(\theta)$ elementwise and $\nabla_\theta \log p(y_k|\theta_k) = \psi_k \big(y_k - \sigma(\psi_k' \theta_k) \big)$.

DATASET

We consider the benchmark adult `a9a` dataset which can be downloaded from `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html`.

The dataset consists of a time series of binary valued (categorical) observations $y_k \in \{0, 1\}$ and a time series of regression vectors $\bar{\psi}_k \in \mathbb{R}^{123}$ for $k = 1, \ldots, 32651$. To model the bias, we add one additional component; so the unknown parameter vector is $\theta \in \mathbb{R}^{124}$ and the augmented regression vectors are $\psi_k = \begin{bmatrix} 1 \\ \bar{\psi}_k \end{bmatrix} \in \mathbb{R}^{124}$, for $k = 1, \ldots, 32651$.

PERFORMANCE OF IRL ALGORITHM (20)

Suppose the inverse learner observes the estimates $\{\theta_k\}$ generated by the RL agents according to (40). The inverse learner aims to reconstruct the posterior $p(\theta | y_{1:T})$. Since $N = 124$, the IRL algorithm needs to explore and sample from a 124-variate distribution which is a formidable task. The vanilla IRL algorithm (2) is not tractable since it would take a prohibitive number of iterations to converge. We illustrate the performance of the multi-kernel variance reduction IRL algorithm (20).

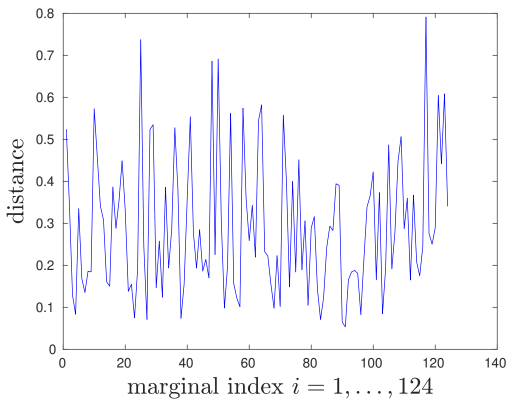
We ran multi-kernel IRL algorithm (20) and active IRL algorithm (27) on the `a9a` dataset. As mentioned in Sec.2.2.3, the active IRL (27) is an idealization of the multikernel IRL algorithm (20) and so forms a benchmark for it. The parameters were chosen as $\mu = 2.5 \times 10^{-4}$, $\pi(\theta) = \mathbf{N}(0, I)$, $\sigma = 0.1$, $L = 100$ in (20) and $T = 10$ in (39). As in Welling and Teh (2011), we ran 10 "sweeps" through the dataset. That is, we appended 9 repetitions of the data set resulting in a single dataset of 10×32651 time points; and then ran the IRL algorithms on this appended dataset.

To benchmark these algorithms, we also ran the classical Langevin dynamics algorithm:

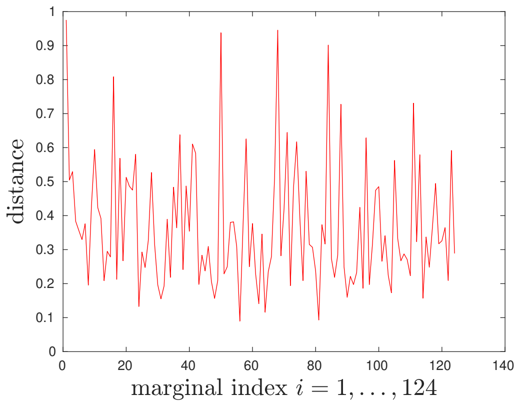
$$\alpha_{k+1} = \alpha_k + \mu \, \frac{\beta}{2} \nabla r_k(\alpha_k) + \sqrt{\mu} \, w_k, \quad k = 1, 2, \ldots, \tag{41}$$

which corresponds to the ground truth (since the gradients are evaluated at α_k).

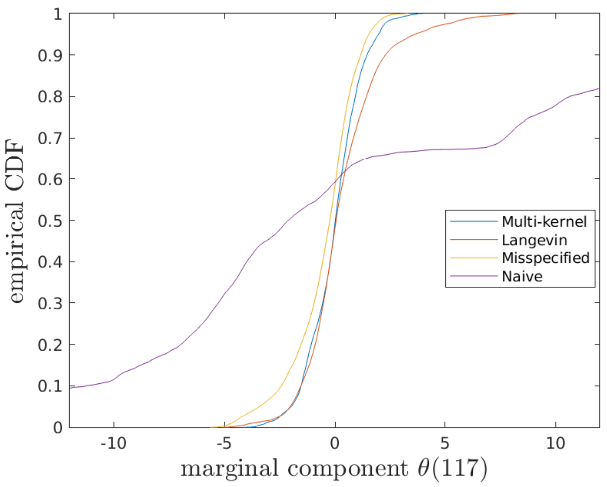
IRL Algorithms (20) and (27) generate samples $\{\alpha_k\}$ from a 124-dimensional distribution. To visualize the performance, we used the output sequence $\{\alpha_k\}$ from these algorithms to compute



(a) Active IRL Algorithm (27) vs ground truth. Wasserstein 1 distance (42) of each of the 124 marginals



(b) Multikernel Algorithm (20) vs ground truth. Wasserstein 1 distance (42) of each of the 124 marginals



(c) Comparison of 117th marginal

Figure 5: Comparison of multi-kernel IRL Algorithm (20) and active IRL algorithm (27) with classical Langevin (41) (ground truth)

the empirical cumulative distribution functions for each of the 124 marginal distributions, denoted by $\hat{F}_i(\alpha(i))$, $i = 1, \dots, 124$. For each such marginal empirical distribution, we then computed the corresponding marginal from the classical Langevin dynamics (41), denoted as $F_i(\alpha(i))$; this can be viewed as the ground truth. Finally, we computed the L_1 distance (Wasserstein 1-metric)

$$d(i) = \int |\hat{F}_i(\alpha(i)) - F_i(\alpha(i))| \, d\alpha(i), \quad i = 1, \dots, 124. \tag{42}$$

This L_1 distance is more appropriate for our purposes than the Kolmogorov-Smirnov distance since typically the constant or proportionality β is not known and so the regions of support of the empirical cumulative distribution functions can vary substantially.

Figure 5(a) and (b) display the L_1 distance $d(i)$ vs $i = 1, 2, \dots, 124$ for the IRL Algorithms (20) and (27). In a sense, Algorithm (27) can be viewed as an upper bound for the performance of

Algorithm (20) since the conditional density $p(\theta|\alpha_k)$ used to generate θ in Algorithm (27) is exactly the same kernel used in Algorithm (20). As can be seen from Figure 5(a) and (b), the two algorithms perform similarly, despite the fact that Algorithm (20) has no control over where the derivative is evaluated. This shows that the IRL algorithm is a viable method for sampling from the high-dimensional Bayesian posterior; or equivalently estimating $J(\theta, \theta^o)$ in (39). Finally, Figure 5(c) shows the marginal distribution for the 117-th component of θ for the classical Langevin (ground truth), active IRL (mis-specified), multi-kernel IRL and a naive Langevin. By naive Langevin we mean the Langevin algorithm that uses the gradient $\nabla_\theta r_k(\theta_k)$ instead of $\nabla_\theta r_k(\alpha_k)$ at the estimate α_k, without any kernel. We see that the multi-kernel and active IRL are close to the ground truth (Langevin) while the naive IRL performs very poorly (since it completely disregards the fact that the gradients evaluated at α_k and θ_k are different).

3.3 Example 3. IRL for Constrained Markov Decision Process (CMDP)

In this section we illustrate the performance of the IRL algorithms for reconstructing the cumulative reward of a constrained Markov decision process (CMDP) given gradient information from a RL algorithm. This is in contrast to classical IRL (Ng and Russell, 2000) where the transition matrices of the MDP are assumed known to the inverse learner.

Consider a unichain[10] average reward CMDP $\{x_n\}$ with finite state space $\mathcal{X} = \{1, \dots, X\}$ and action space $\mathcal{U} = \{1, 2, \dots, U\}$. The CMDP evolves with transition probability matrix $P(u)$ where

$$P_{ij}(u) \stackrel{\triangle}{=} \mathbb{P}[x_{n+1} = j | x_n = i, u_n = u], \quad u \in \mathcal{U}. \tag{43}$$

When the system is in state $x_n \in \mathcal{X}$, an action $u_n = \mathbf{u}(x_n) \in \mathcal{U}$ is chosen, where $\mathbf{u}$ denotes (a possible randomized) stationary policy. The reward incurred at stage n is $\rho(x_n, u_n) \geq 0$.

Let $\mathcal{D}$ denote the class of stationary randomized Markovian policies. For any stationary policy $\mathbf{u} \in \mathcal{D}$, let $\mathbb{E}_{\mathbf{u}}$ denote the corresponding expectation and define the infinite horizon average reward

$$J(\mathbf{u}) = \lim_{T \to \infty} \inf \frac{1}{T} \mathbb{E}_{\mathbf{u}} \Big[\sum_{n=1}^{T} \rho(x_n, u_n) \mid x_0 = x \Big]. \tag{44}$$

Motivated by modeling fairness constraints in network optimization (Ngo and Krishnamurthy, 2010), we consider the reward (44), subject to the average constraint:

$$B(\mathbf{u}) = \lim_{T \to \infty} \inf \frac{1}{T} \mathbb{E}_{\mathbf{u}} \Big[\sum_{n=1}^{T} \beta(x_n, u_n) \Big] \leq \gamma, \tag{45}$$

(44), (45) constitute a CMDP. Solving a CMDP involves computing the optimal policy $\mathbf{u}^* \in \mathcal{D}$ that satisfies

$$J(\mathbf{u}^*) = \sup_{\mathbf{u} \in \mathcal{D}} J(\mathbf{u}) \quad \forall x_0 \in \mathcal{X}, \text{ subject to } (45) \tag{46}$$

To solve a CMDP, it is sufficient to consider randomized stationary policies:

$$\mathbf{u}(x) = u \text{ with probability } \phi(u|x) = \frac{\bar{\phi}(x, u)}{\sum_{\tilde{u} \in \mathcal{U}} \bar{\phi}(x, \tilde{u})}, \tag{47}$$

10. By *unichain* (Puterman, 1994, pp. 348) we mean that every policy where u_n is a deterministic function of x_n consists of a single recurrent class plus possibly an empty set of transient states.

where the conditional probabilities ϕ and joint probabilities $\bar{\phi}$ are defined as

$$\phi(u|x) = \mathbb{P}(u_n = u|x_n = x), \quad \bar{\phi}(x,u) = \mathbb{P}(u,x). \tag{48}$$

Then the optimal policy $\mathbf{u}^*$ is obtained as the solution of a linear programming problem in terms of the $X \times U$ elements of $\bar{\phi}$; see Puterman (1994) for the precise equations.

Also (Altman, 1999), the optimal policy $\mathbf{u}^*$ of the CMDP is *randomized* for at most one of the states. That is,

$$\mathbf{u}^*(x) = p\,\mathbf{u}_1^*(x) + (1-p)\,\mathbf{u}_2^*(x) \tag{49}$$

where $p \in [0,1]$ denotes the randomization probability and $\mathbf{u}_1^*, \mathbf{u}_2^*$ are pure (non-randomized) policies. Of course, when there is no constraint (45), the CMDP reduces to classical MDP and the optimal stationary policy $\mathbf{u}^*(x)$ is a pure policy. That is, for each state $x \in \mathcal{X}$, there exists an action u such that $\phi(u|x) = 1$.

Remarks. (i) (45) is a global constraint that applies to the entire sample path (Altman, 1999). Since the optimal policy is randomized, classical value iteration based approaches and Q-learning cannot be used to solve CMDPs as they yield deterministic policies. One can construct a Lagrangian dynamic programming formulation (Altman, 1999) and Lagrangian Q-learning algorithms (Djonin and Krishnamurthy, 2007). Below for brevity, we consider a policy gradient RL algorithm.

(ii) *Discounted CMDPs*. Instead of an average cost CMDP, a discounted cost CMDP can be considered. Discounted CMDPs are less technical in the sense that an optimal policy always exists (providing the constraint set is non-empty); whereas average cost CMDPs require a unichain assumption. It is easily shown (Krishnamurthy, 2016) that the dual linear program of a discounted CMDP can be expressed in terms of the conditional probabilities $\phi(u|x)$ and the optimal randomized policy is of the form (49). The final IRL algorithm is identical to (54) below.

3.3.1 POLICY GRADIENT FOR RL OF CMDP

Having specified the CMDP model, we next turn to the RL algorithm. RL algorithms[11] are used to estimate the optimal policy of an MDP when the transition matrices are not known. Then the LP formulation in terms of joint probabilities $\bar{\phi}$ is not useful since the constraints depend on the transition matrix. In comparison, *policy gradient RL algorithms* are stochastic gradient algorithms of the form (1) that operate on the conditional action probabilities $\phi(u|x)$ defined in (48) instead of the joint probabilities $\bar{\phi}(x,u)$.

Note that (46) written as a minimization (in terms of $-J$), together with constraint (45) is in general, no longer a convex optimization problem in the variables ϕ; see Figure 6 for an illustration. So it is not possible to guarantee that simple gradient descent schemes[12] can achieve the global optimal policy. This motivates the setting of (1) where multiple agents that are initialized randomly aim to estimate the optimal policy.

Since the problem is non-convex, and the inequality constraint is active (i.e., achieves equality) at the global maximum, we assume that the RL agents use a quadratic penalty method: For $\lambda \geq 0$,

11. In adaptive control, RL algorithms such as policy gradient are viewed as simulation based *implicit* adaptive control methods that bypass estimating the MDP parameters (transition probabilities) and directly estimate the optimal policy.
12. Consider minimizing the negative of the objective function, namely $-J$ without constraint (45). Even though $-J$ is nonconvex in ϕ, one can show (using Lyapunov function arguments) that for this unconstrained MDP case, the gradient algorithm will converge to a global optimum. However for the constrained MDP case this is not true; the nonconvex objective and constraints results in a duality gap.

denote the quadratic penalized objective to be maximized as

$$R(\phi) = J(\phi) - \lambda\,(B^2(\phi) - \gamma) \tag{50}$$

Such quadratic penalty functions are used widely for equality constrained non-convex problems.

The RL agents aim to minimize the T-horizon sample path penalized objective which at batch k is

$$\begin{aligned} r_k(\phi) &\overset{\triangle}{=} J_{k,T}(\phi) + \lambda\left(B^2_{k,T}(\phi) - \gamma\right), \quad \lambda \in \mathbb{R}_+ \\ J_{k,T} &= \frac{1}{T}\sum_{n=1}^{T} \rho(x_n, \mathbf{u}_\phi(u_n)), \quad B_{k,T} = \frac{1}{T}\sum_{n=1}^{T} \beta(x_n, \mathbf{u}_\phi(u_n)) \end{aligned} \tag{51}$$

There are several methods for estimating the policy gradient $\nabla_\phi r_k(\phi_k)$ (Pflug, 1996) including the score function method, weak derivatives (Abad and Krishnamurthy, 2003) and finite difference methods. A useful finite difference gradient estimate is given by the SPSA algorithm (Spall, 2003); useful because SPSA evaluates the gradient along a single random direction.

3.3.2 IRL FOR CMDP

Consider the CMDP (43), (44), (47). Assume we are given a sequence of gradient estimates $\{\nabla_\phi r_k(\phi_k)\}$ of the sample path wrt to the parametrized policy ϕ from (51). The aim of the inverse learner is to reconstruct the reward $R(\phi)$ in (50). Since by construction the constraint is active at the optimal policy, the aim of the inverse learner is to explore regions of ϕ in the vicinity where the constraint $\{\phi : B(\phi) \approx \gamma\}$ is active in order to estimate $R(\phi)$.

A naive application of Langevin IRL algorithm (2) to update the conditional probabilities $\{\phi_k\}$ will not work. This is because there is no guarantee that the estimate sequence $\{\phi_k\}$ generated by the algorithm are valid probability vectors, namely

$$\phi_k(u|x) \in [0,1], \quad \sum_{u\in\mathcal{U}} \phi_k(u|x) = 1, \quad x \in \mathcal{X}. \tag{52}$$

We will use spherical coordinates[13] to ensure that the conditional probability estimates ϕ_k generated by the IRL algorithm satisfies (52) at each iteration k. The idea is to parametrize $\sqrt{\phi_k(u|x)}$ to lie on the unit hyper-sphere in $\mathbb{R}^U$. Then all needed are the $U-1$ angles for each x, denoted as $\theta(i,1), \ldots \theta(i,U-1)$. Define the spherical coordinates in terms of the mapping:

$$\phi = \mathcal{E}(\theta), \qquad \text{where } \phi(u|x) = \begin{cases} \cos^2\theta(i,1) & \text{if } u = 1 \\ \cos^2\theta(i,u)\prod_{p=1}^{u-1}\sin^2\theta(i,p) & u \in \{2,\ldots,U-1\} \\ \sin^2\theta(i,U-1)\prod_{p=1}^{U-2}\sin^2\theta(i,p) & u = U \end{cases} \tag{53}$$

Then clearly $\phi(u|x)$ in (53) always satisfies feasibility (52) for any real-valued (un-constrained) $\theta(x,u)$. To summarize, there are $(U-1)X$ unconstrained parameters in θ. Also for $\theta(i,u) \in$

13. Another parametrization widely used in machine learning is exponential coordinates: $\phi(u|x) = \frac{\exp(\theta(x,u))}{\sum_{a\in\mathcal{U}}\exp(\theta(x,a))}$, where $\theta(x,u) \in \mathbb{R}$ is unconstrained. However, as shown in Krishnamurthy (2016); Krishnamurthy and Vazquez Abad (2018), spherical coordinates typically yield faster convergence. We also found this in numerical studies on IRL (not presented here).

$[0, \pi/2]$, the mapping $\mathcal{E} : \mathbb{R}^{U \times X} \to \mathbb{R}^{U \times X}$ in (53) is one-to-one and therefore invertible. We denote the inverse as $\mathcal{E}^{-1}$.

Remark: As an example, consider $U = 2$. Then in spherical coordinates $\phi(1|i) = \sin^2 \theta(i, 1)$, $\phi(2|i) = \cos^2 \theta(i, 1)$, where $\theta(i, 1)$ is un-constrained.; clearly $\phi(1|i) + \phi(2|i) = 1$, $\phi(u|i) \geq 0$.

With the above re-parametrization, we can run any of the passive Langevin dynamics IRL algorithms proposed in this paper. In the numerical example below, we ran the two-time scale multi-kernel IRL algorithm (20). Recall this does not require knowledge of $\pi(\cdot)$ and also provides variances reduction: Given the current IRL estimate α_k, the RL gives us a sequence $\{\phi_i, \nabla_\phi r_k(\phi_i), i = 1, \ldots, L\}$ The IRL algorithm (20) operating on the $(U - 1)X$ unconstrained parameters of θ is:

$$\begin{aligned}
\alpha_{k+1} &= \alpha_k + \mu \, \frac{\beta}{2} \, \frac{\sum_{i=1}^L p(\theta_i|\alpha_k) \nabla_\theta r_k(\theta_i)}{\sum_{l=1}^L p(\theta_l|\alpha_k)} + \sqrt{\mu} w_k, \qquad \phi_i \sim \pi(\cdot) \\
\text{where} \quad \theta_i &= \mathcal{E}^{-1}(\phi_i), \quad \nabla_\theta r_k(\theta_i) = (\nabla_\phi r_k(\phi_i))' \, \nabla_\theta \phi_i, \\
p(\theta|\alpha) &= p_v(\theta - \alpha), \qquad p_v(\cdot) = \mathbf{N}(0, \sigma^2 I_N)
\end{aligned} \tag{54}$$

In the second line of (54), we transformed $\nabla_\phi r_k(\phi_k)$ to $\nabla_\theta r_k(\theta_k)$ to use in the IRL algorithm.

To summarize, the IRL algorithm (54) generates samples $\alpha_k \sim \exp(R(\mathcal{E}(\alpha)))$. Equivalently, $\phi_k = \mathcal{E}(\alpha_k) \sim \exp(R(\phi))$, where $R(\phi)$ is defined in (50). Thus given only gradient information from a RL algorithm, we can reconstruct (sample from) the penalized reward $R(\cdot)$ of the CMDP without any knowledge of the CMDP parameters.

3.3.3 NUMERICAL EXAMPLE

We generated a CMDP with $X = 2$ (2 states), $U = 2$ (2 actions) and 1 constraint with

$$P(1) = \begin{bmatrix} 0.8 & 0.2 \\ 0.3 & 0.7 \end{bmatrix}, \quad P(2) = \begin{bmatrix} 0.6 & 0.4 \\ 0.1 & 0.9 \end{bmatrix}, \rho = \begin{bmatrix} 1 & 100 \\ 30 & 2 \end{bmatrix}, \beta = \begin{bmatrix} 0.2 & 0.3 \\ 2 & 1 \end{bmatrix}, \gamma = 1, \lambda = 10^5 \tag{55}$$

Recall the transition matrices $P(u)$ are defined in (43), the reward matrix $(\rho(x, u))$ in (44), constraint matrix $(\beta(x, u))$ and γ in (45), and penalty multiplier λ in (51).

The randomized policy $\phi(u|x)$, $u \in \{1, 2\}$, $x \in \{1, 2\}$ is a 2×2 matrix. It is completely determined by $(\phi(1|1), \phi(1|2)) \in [0, 1] \times [0, 1]$; so it suffices to estimate $R(\phi)$ over $[0, 1] \times [0, 1]$.

Figure 6(a) displays the cumulative reward $J(\phi)$; this constitutes the ground truth. To obtain this figure, we computed the average reward MDP value function $J(\phi)$ and constraint $B(\phi)$ for each policy ϕ where ϕ sweeps over $[0, 1] \times [0, 1]$. Given a policy ϕ, $J(\phi)$ and $B(\phi)$ are computed by first evaluating the joint probability $\bar{\phi}$ as (Ross, 1983, pp.101)

$$\bar{\phi}(j, a) = \sum_i \sum_{\bar{a}} \bar{\phi}(i, \bar{a}) \, P_{ij}(\bar{a}) \, \phi(a|j), \quad \sum_j \sum_a \bar{\phi}(j, a) = 1$$

and then $J(\phi) = \sum_x \sum_u \bar{\phi}(x, u) \rho(x, u)$, $B(\phi) = \sum_x \sum_u \bar{\phi}(x, u) \beta(x, u)$.

For values of ϕ that do not satisfy the constraint $B(\phi) < \gamma$, we plot $J(\phi) = 0$. Figure 6(a) illustrates the non-convex nature of the constraint set.

Figure 6(b) displays the penalized cumulative reward $R(\phi) = J(\phi) - \lambda \, (B(\phi) - \gamma)^2$ where the quadratic penalty function is $\lambda \, (B(\phi) - \gamma)^2$. As mentioned earlier, since we know that the constraint is active at the optimal policy, we want the IRL to explore the vicinity of the region of ϕ where the constraint is active.

We then ran the IRL algorithm (54) using spherical coordinates with parameters $\mu = 5 \times 10^{-6}$, $\sigma = 0.1$, $L = 50$ for $T = 10^5$ iterations. Figure 6(c) displays a 3-dimensional stem plots of the log of the empirical distribution of $\phi_k = \mathcal{E}(\alpha_k)$. wrt coordinates $\phi(1|1)$ and $\phi(1|2)$. As can be seen from the two plots, the IRL algorithm samples from the high probability regions $\{\phi : B(\phi) \approx \gamma\}$ to reconstruct the penalized reward $R(\phi)$. Specifically, the C-shaped curve profile generated by the IRL estimates match the C-shaped curve of the penalized cumulative reward Figure 6(b).

4. Weak Convergence Analysis of IRL Algorithm

This section discusses the main assumptions, weak convergence theorem and proof regarding IRL algorithm (2). (Recall the informal proof in Sec.2.1 for the motivation of weak convergence.)

NOTATION

- Since $\nabla_\theta r_k(\theta_k)$ is a noise corrupted estimate of the gradient $\nabla_\theta R(\theta)$, we write it in more explicit notation as $\widetilde{r}(\theta_k, \xi_k)$, where $\{\xi_k\}$ is a sequence of random variables satisfying appropriate conditions specified below.
- We use $\pi_\alpha(\cdot)$ to denote $\nabla_\alpha \pi(\cdot)$.
- Finally, $\mathbb{E}_m$ denotes the conditional expectation (conditioning up to time m), i.e., conditioning wrt the σ-algebra $\mathcal{F}_m = \sigma\{\alpha_0, \theta_j, \xi_j;\ j < m\}$.

ALGORITHM

There are two possible implementations of IRL algorithm (2). The first implementation is (2), namely,

$$\alpha_{k+1} = \alpha_k + \frac{\mu}{\Delta^N} K\Big(\frac{\theta_k - \alpha_k}{\Delta}\Big) \frac{\beta}{2} \widetilde{r}(\theta_k, \xi_k) \pi(\alpha_k) + \mu \pi_\alpha(\alpha_k) \pi(\alpha_k) + \sqrt{\mu}\, \pi(\alpha_k) w_k, \tag{56}$$

and the second implementation is

$$\alpha_{k+1} = \alpha_k + \frac{\mu}{\Delta^N} K\Big(\frac{\theta_k - \alpha_k}{\Delta}\Big) \Big[\frac{\beta}{2} \widetilde{r}(\theta_k, \xi_k) \pi(\theta_k) + \pi_\alpha(\theta_k)\Big] + \sqrt{\frac{\mu}{\Delta^N}} K\Big(\frac{\theta_k - \alpha_k}{\Delta}\Big) \pi(\theta_k) w_k, \tag{57}$$

where μ is the stepsize and $\Delta = \Delta(\mu)$ is chosen so $\mu/\Delta^N \to 0$ as $\mu \to 0$.

Both the above algorithms converge to the same limit. The proof below is devoted to (56), but (57) can be handled similarly. Also the proofs of the other two proposed IRL algorithms, namely (17) and (29) are similar.

Taking a continuous-time interpolation

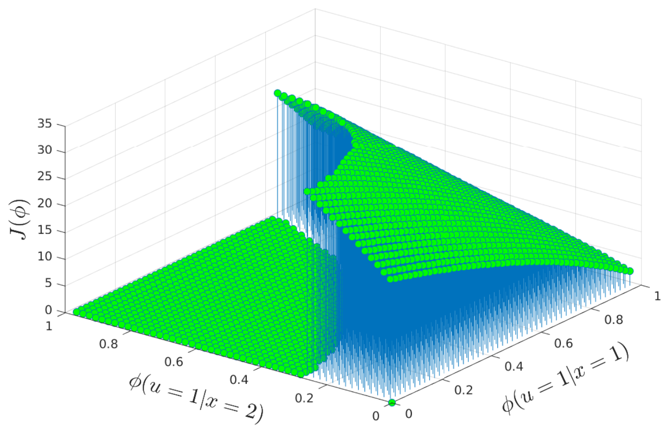
$$\alpha^\mu(t) = \alpha_k \ \text{ for } \ t \in [\mu k, \mu k + \mu), \tag{58}$$

we aim to show that the sequence $\alpha^\mu(\cdot)$ converges weakly to $\alpha(\cdot)$, which give the desired limit.

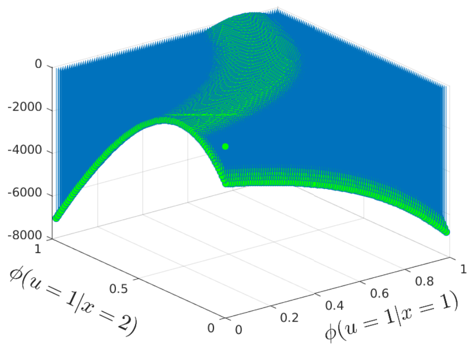
4.1 Assumptions

We begin by stating the conditions needed.

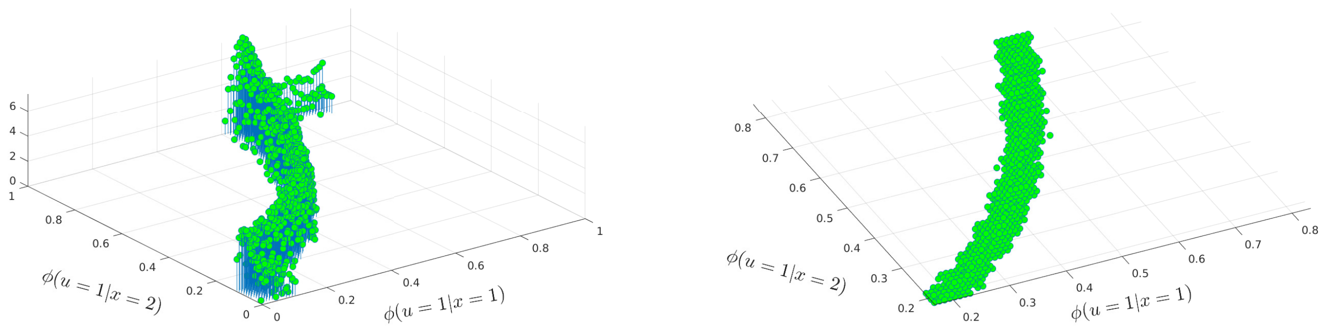
(A1) For each ξ, $\widetilde{r}(\cdot, \xi)$ has continuous partial derivatives up to the second order such that the second partial $\widetilde{r}_{\alpha\alpha}(\cdot, \xi)$ is bounded. For each $b < \infty$ and $T < \infty$, $\{\widetilde{r}(\alpha, \xi_j); |\alpha| \le b, j\mu \le T\}$ is uniformly integrable.



(a) Cumulative Reward $J(\phi)$ with active constraint $B(\phi) \leq 1$. The non-convexity of the constraint set is clearly seen.



(b) Penalized Cumulative Reward with Quadratic Penalty $R(\phi) = J(\phi) - \lambda\,(B(\phi)-1)^2$. The lighter green shade on top shows the active constraint. This plot constitutes the ground truth



(c) IRL algorithm estimate. Snapshot 1 shows that the IRL estimates $R(\phi)$ in the vicinity of the active constraint.. Snapshot 2 shows that the IRL explores regions in the vicinity of the active constraint. Specifically the curve is close to the lighter shade green in Fig (b)

Figure 6: IRL for Constrained MDP

(A2) The sequences $\{\theta_k\}$ is stationary and independent of $\{\xi_k\}$. For each $k \geq n$, there exists a conditional density of θ_k given $\mathcal{F}_n$, denoted by $\pi_k(\theta|\mathcal{F}_n)$ such that $\pi_k(\theta|\mathcal{F}_n) > 0$ for each θ and that $\pi_k(\cdot|\mathcal{F}_n)$ is continuous. The sequence $\{\pi_k(\cdot|\mathcal{F}_n)\}_{k\geq n}$ is bounded uniformly. The probability density $\pi(\cdot)$ is continuous and bounded with $\pi(\theta) > 0$ for each θ such that

$$\lim_{k-n\to\infty} \mathbb{E}|\pi_k(\theta|\mathcal{F}_n) - \pi(\theta)| = 0. \tag{59}$$

(A3) The measurement noise $\{\xi_n\}$ is exogenous, and bounded stationary mixing process with mixing measure φ_k such that $\mathbb{E}\widetilde{r}(\alpha, \xi_k) = R_\alpha(\alpha)$ for each α and $\sum_k \varphi_k < \infty$. The $\{w_k\}$ is a sequence of $\mathbb{R}^N$-valued i.i.d. random variables with mean 0 and covariance matrix I (the identity matrix); $\{w_k\}$ and $\{\xi_k\}$ are independent.

(A4) The kernel $K(\cdot)$ satisfies

$$\begin{aligned} &K(u) \geq 0,\ K(u) = K(-u), \sup_u K(u) < \infty, \\ &\int K(u)du = 1,\ \int |u|^2 K(u)du < \infty. \end{aligned} \tag{60}$$

Remarks: We briefly comment on the assumptions (A1)-(A4).

- Assumption (A1) requires the smoothness of $\widetilde{r}(\cdot, \xi)$, which is natural because we are using $\widetilde{r}(\cdot, \xi_k)$ to approximate the smooth function ∇R. We consider a general noise so the uniform integrability is used. If the noise is additive in that $\widetilde{r}(\theta, \xi) = \nabla R(\theta) + \xi$, then we only need the finite $\widetilde{p}$-moments of ξ_k for $\widetilde{p} > 1$.
- Assumption (A3) requires the stochastic process $\{\xi_n\}$ to be exogenous, and bounded stationary mixing. Thus for each α, $\{\widetilde{r}(\alpha, \xi_k)\}$ is also a mixing sequence. A mixing process is one in which remote past and distant future are asymptotically independent. It covers a wide range of random processes such as i.i.d. sequences, martingale difference sequences, moving average sequences driving by a martingale difference sequence, and functions of stationary Markov processes with a finite state space (Billingsley, 1999), etc. The case of $\{w_k\}$ and $\{\xi_k\}$ being dependent can be handled, but for us $\{w_k\}$ is the added perturbation to get the desired Brownian motion so independence is sufficient.
- By exogenous in (A3), we mean that

$$\begin{aligned} &P(\xi_{n+1} \in A_1, \ldots, \xi_{n+k} \in A_k | \alpha_0, \xi_j, x_j;\ j \leq n) \\ &\quad = P(\xi_{n+1} \in A_1, \ldots, \xi_{n+k} \in A_k | \alpha_0, x_j, \xi_j, \alpha_{j+1};\ j \leq n), \end{aligned}$$

  for all Borel sets A_i, $i \leq k$, and for all k and n.
- In view of the mixing condition (A3) on $\{\xi_k\}$, for each $b < \infty$ and $T < \infty$, $\{\widetilde{r}(\alpha, \xi_j); |\alpha| \leq b, j\mu \leq T\}$ and $\{\widetilde{r}_\alpha(\alpha, \xi_j); |\alpha| \leq b, j\mu \leq T\}$ are uniformly integrable.
- Again, using the mixing condition, for each α, as $n \to \infty$,

$$\frac{1}{n} \sum_{j=m}^{m+n-1} \mathbb{E}_m \widetilde{r}(\alpha, \xi_j) \to R_\alpha(\alpha) \text{ in probability.} \tag{61}$$

- For a Borel set A, we have $P(\theta_k \in A|\mathcal{F}_n) = \int_{\theta\in A} \pi_k(\theta|\mathcal{F}_n)d\theta$. If $\{\theta_n\}$ is itself a stationary ϕ-mixing sequence with a continuous density, and if $\mathbb{E}|\theta_n|^2 < \infty$, then by virtue of a well-known mixing inequality, some $\widetilde{c}_0 > 0$, (Ethier and Kurtz, 1986, Corollary 2.4 in Chapter 7),

$$\mathbb{E}\{|\int \theta\pi_k(\theta|\mathcal{F}_n)d\theta - \int \theta\pi(\theta)d\theta|\} \leq \widetilde{c}_0 \varphi_\theta^{1/2}(k-n)\mathbb{E}^{1/2}|\theta_k|^2 \to 0 \text{ as } k - n \to \infty,$$

where $\varphi_\theta(\cdot)$ denotes the mixing measure.

- Condition (A4) is concerned with the properties of $K(\cdot)$. It assumes that the kernel is nonnegative, symmetric, bounded (similar to a probability density function), and square integrable. (A4) is satisfied by a large class of kernels. For example, commonly used symmetric kernels with compact supports satisfy this condition (e.g., truncated Gaussian kernels). Moreover, it is also verifiable for kernels with *unbounded* support. A crucial point is that the tails of $K(\cdot)$ are small (asymptotically negligible). For simplicity, we use (A4) as a nicely packaged version. In fact, (A4) is a sufficient condition for a much larger class of kernels satisfying

$$\int K(u)du = 1, \int |u|^l K(u)du < \infty \text{ some } l, \tag{62a}$$

$$\int K^2(u)du < \infty, \int |u|^2 K(u)du < \infty, \tag{62b}$$

$$\int (u^1)^{m_1}(u^2)^{m_2}\cdots(u^N)^{m_N}K(u)du = 0 \text{ if } l > 1, \tag{62c}$$

$$\text{where } 1 \le m_1 + m_2 + \cdots + m_N \le l-1 \tag{62d}$$

Here $u^1, \dots, u^N$ denote the components of $u \in \mathbb{R}^N$. The parameter l is a smoothness indicator of the kernel and the last line of (62) is often used in nonparametric estimation in statistics. Such a condition stems from a large class of kernels used in the so-called lth-order averaging operator; see Katkovnik (1976). Thus, (A4) can be replaced by this more general setup. However, we use the current form of (A4) because it is easily verifiable (e.g., by Gaussian kernel).

Eq.(62) can be written in multi-index notation as follows. Let $m = (m_1, \dots, m_N)$ where each m_i is a nonnegative integer, $|m| = \sum_{i=1}^N m_i$, and $m! = \prod_{i=1}^N m_i!$. Thus, $u^m = (u^1)^{m_1}\cdots(u^N)^{m_N}$. Then (62c), (62d) can be written in multi-index notation as

$$\int u^m K(u)du = 0 \text{ if } 1 \le |m| \le l-1 \text{ and } l > 1.$$

4.2 Main Result and Proof

As is well known (Kushner and Yin, 2003), a classical fixed step size stochastic gradient algorithm converges weakly to a *deterministic* ordinary differential equation (ODE) limit; this is the basis of the so called ODE approach for analyzing stochastic gradient algorithms. In comparison, the discrete time IRL algorithm (2) converges weakly to a *stochastic* process limit $\alpha(\cdot)$. In this section we prove weak convergence of the interpolated process $\{\alpha^\mu(\cdot)\}$ to the stochastic process limit $\alpha(\cdot)$ as $\mu \to 0$. Proving weak convergence requires first that the tightness of the sequence be verified and then the limit be characterized via the so called martingale problem formulation. For a comprehensive treatment of the martingale problem of Stroock and Varadhan, see Ethier and Kurtz (1986).

Theorem 1 *Assume conditions* (A1)-(A4). *Then the interpolated process* $\alpha^\mu(\cdot)$ (*defined in* (58)) *for IRL algorithm* (2) *has the following properties:*

1. $\{\alpha^\mu(\cdot)\}$ *is tight in* $D^d[0,\infty)$.
2. *Any weakly convergent subsequence of* $\{\alpha^\mu(\cdot)\}$ *has a limit* $\alpha(\cdot)$ *that satisfies*

$$\begin{aligned} d\alpha(t) &= \Big[\frac{\beta}{2}\pi^2(\alpha(t))R_\alpha(\alpha(t)) + \pi_\alpha(\alpha(t))\pi(\alpha(t))\Big]dt + \pi(\alpha(t))dW(t), \\ \alpha(0) &= \alpha_0, \end{aligned} \tag{63}$$

where $W(\cdot)$ is a standard Brownian motion with mean 0 and covariance being the identity matrix $I \in \mathbb{R}^{N\times N}$, provided (63) *has a unique weak solution* (*in a distributional sense*) *for each initial condition.*

For sufficient conditions leading to unique weak solutions of stochastic differential equation and uniqueness of martingale problem, see Ethier and Kurtz (1986, p. 182) or Karatzas and Shreve (1991).

Proof. The proof is divided into 4 steps.

Step 1. Use a truncation device. Because the sequence $\{\alpha_k\}$ is not *a priori* bounded, the main idea is to use a truncation device (Kushner and Yin, 2003, p.284). (Step 4 below deals with the un-truncated process.) Let $M > 0$ be a fixed but otherwise arbitrary constant. Denote by $S_M = \{\alpha \in \mathbb{R}^N : |\alpha| \le M\}$ the N-dimensional ball centered at the origin with radius M. Consider the truncated algorithm

$$\alpha^M_{k+1} = \alpha^M_k + \mu\Big[\frac{1}{\Delta^N}K\Big(\frac{\theta_k - \alpha^M_k}{\Delta}\Big)\frac{\beta}{2}\widetilde{r}(\theta_k,\xi_k) + \pi_\alpha(\alpha^M_k)\Big]\pi(\alpha^M_k)q_M(\alpha^M_k) + \sqrt{\mu}\pi(\alpha^M_k)q_M(\alpha^M_k)w_k, \tag{64}$$

where

$$q_M(\alpha) = \begin{cases} 1, \ \alpha \in S_M; \\ 0, \ \alpha \in \mathbb{R}^N - S_{M+1}; \\ \text{smooth} \quad \text{otherwise.} \end{cases}$$

By virtue of (A4), the integrability of the kernel forces θ_k to be in line with the iterates α^M_k in that only asymptotically negligible tails can be added.

Remark. Define $\alpha^{\mu,M}(t) = \alpha^M_n$ on $[\mu k, \mu k + \mu)$. Then $\alpha^{\mu,M}(\cdot) \in D^N[0,\infty)$ and is an M-truncation for $\alpha^\mu(\cdot)$ (Kushner and Yin, 2003, p.284). We proceed to prove the tightness and weak convergence of the truncated sequence $\{\alpha^{\mu,M}(\cdot)\}$ first and then complete the proof by letting $M \to \infty$ in Step 4.

Step 2. Prove the tightness of $\{\alpha^{\mu,M}(\cdot)\}$. Note that in view of Nazin et al. (1989, Lemma 1), by virtue of (A4), for a function $h(\cdot)$ that is twice continuously differentiable with bounded second derivative, it follows that

$$\Big|\frac{1}{\Delta^N}\int K\Big(\frac{\theta-\alpha}{\Delta}\Big)h(\theta)d\theta - h(\alpha)\Big| = O(\Delta^2). \tag{65}$$

Using (65), (A1), and noting that $\{w_k\}$ is an i.i.d. sequence with mean 0 and covariance matrix I, we can show that $\Big\{\big[\Delta^{-N}K\Big(\frac{\theta_k-\alpha^M_k}{\Delta}\Big)\frac{\beta}{2}\widetilde{r}(\theta_k,\xi_k) + \pi_\alpha(\alpha^M_k)\big]\pi(\alpha^M_k)q_M(\alpha^M_k)\Big\}$ is uniformly integrable and also $\{\pi(\alpha^M_k)q_M(\alpha^M_k)w_k\}$ is uniformly integrable. Then using Kushner (1984, p.51, Lemma 7) (or use a perturbed test function methods as in Kushner and Yin (2003, Chapter 7)), it can be shown that $\{\alpha^{\mu,M}(\cdot)\}$ is tight in $D([0,\infty),\mathbb{R}^N)$, the space of $\mathbb{R}^N$-valued functions that are right continuous, have left limits, endowed with the Skorohod topology.

Step 3. Characterize the limit process. Because $\{\alpha^{\mu,M}(\cdot)\}$ is tight, by virtue of Prohorov's theorem (Billingsley, 1999), we can extract a weakly convergent subsequence. To simplify notation, still denote the subsequence by $\{\alpha^{\mu,M}(\cdot)\}$ whose limit is $\alpha^M(\cdot)$. By Skorohod representation (Kushner and Yin, 2003, p. 230) with a slight abuse of notation, we may assume that $\alpha^{\mu,M}(\cdot)$ converges to $\alpha^M(\cdot)$ w.p.1. To complete the proof, we need only characterize the limit process by showing that the limit $\alpha^M(\cdot)$ is a solution of the martingale problem with backward operator

$$\mathcal{L}^M f(\alpha) = f'_\alpha(\alpha)\Big[\frac{\beta}{2}\pi^2(\alpha)R_\alpha(\alpha) + \pi_\alpha(\alpha)\pi(\alpha)\Big]q_M(\alpha) + \frac{1}{2}\pi^2(\alpha)\operatorname{Tr}[f_{\alpha\alpha}(\alpha)]q_M(\alpha) \tag{66}$$

for any real-valued function $f(\cdot) \in C_0^2$ (Ethier and Kurtz (1986, Lemma 8.1, p.225)), where f' denotes the transpose of f.

By Theorem 8.2 in Ethier and Kurtz (1986), to verify the martingale property, we need to show that for any bounded and continuous test function $g(\cdot)$, any $t, s > 0$, any positive integer κ_1, and any $t_\imath \le t$,

$$\mathbb{E}\Big\{g(\alpha^M(t_\imath) : \imath \le \kappa_1)\Big[f(\alpha^M(t+s)) - f(\alpha^M(t)) - \int_t^{t+s} \mathcal{L}^M f(\alpha^M(u))du\Big]\Big\} = 0. \tag{67}$$

Note that (67), namely, the solution of the martingale problem, is a statement about the finite dimensional distributions of $\alpha^M(\cdot)$ at times $t_1, \ldots, t_{\kappa_1}$.

To verify (67), we work with the sequence indexed by μ. By the continuity of $f(\cdot)$, the weak convergence, and the Skorohod representation, we have that as $\mu \to 0$,

$$\begin{aligned} &\mathbb{E}g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)[f(\alpha^{\mu,M}(t+s)) - f(\alpha^{\mu,M}(t))] \\ &\quad\to \mathbb{E}g(\alpha^M(t_\imath) : \imath \le \kappa_1)[f(\alpha^M(t+s)) - f(\alpha^M(t))]. \end{aligned} \tag{68}$$

To simplify notation, we denote $q_k^M = q_M(\alpha_k^M)$ in what follows whenever there is no confusion and retain the notation $q_M(\alpha_k^M)$ whenever it is needed. Dividing the segment

$$\lfloor t/\mu \rfloor \le k \le \lfloor (t+s)/\mu \rfloor$$

into sub-blocks of size m_μ each so that $m_\mu \to \infty$ as $\mu \to 0$ and $\delta_\mu = \mu m_\mu \to 0$. Then we obtain

$$\begin{aligned} &\mathbb{E}g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)[f(\alpha^{\mu,M}(t+s)) - f(\alpha^{\mu,M}(t))] \\ &= \mathbb{E}g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)\Big(\mathbb{E}_{lm_\mu}\sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu}[f(\alpha^M_{lm_\mu+m_\mu}) - f(\alpha^M_{lm_\mu})]\Big) \\ &= \mathbb{E}g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)\Big\{\mathbb{E}_{lm_\mu}\sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu} f'_\alpha(\alpha^M_{lm_\mu})\sum_{k=lm_\mu}^{lm_\mu+m_\mu-1}\Big[\frac{\mu}{\Delta^N}K\Big(\frac{\theta_k - \alpha_k^M}{\Delta}\Big) \\ &\qquad\qquad \times\frac{\beta}{2}\widetilde{r}(\theta_k,\xi_k)\pi(\alpha_k^M) + \mu\pi_\alpha(\alpha_k^M)\pi(\alpha_k^M) + \sqrt{\mu}\pi(\alpha_k^M)w_k\Big]q_k^M \\ &\qquad\qquad +\frac{1}{2}\mathbb{E}_{lm_\mu}\mu\sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu}\sum_{k=lm_\mu}^{lm_\mu+m_\mu-1}\pi^2(\alpha_k)\operatorname{Tr}[f_{\alpha\alpha}(\alpha^M_{lm_\mu})w_k w_k']q_k^M \\ &\qquad\qquad +\mathbb{E}_{lm_\mu}\sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu} e_l^\mu\Big\}, \end{aligned} \tag{69}$$

where $\mathbb{E}_{lm_\mu}$ denotes the conditional expectation with respect to the past information up to the time lm_μ (i.e., the σ-algebra generated by $\{\alpha_k, \theta_k, \xi_k : k < lm_\mu\}$), and e_l^μ is an error term. It can be shown that

$$\mathbb{E}g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)\Big|\mathbb{E}_{lm_\mu}\sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu} e_l^\mu\Big|^2 \to 0 \text{ as } \mu \to 0. \tag{70}$$

Noting that $\{w_k\}$ is an i.i.d. sequence with mean 0 and covariance I (the identity matrix), using the continuity of $\pi(\cdot)$, the limit of

$$\mathbb{E}g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)\Big\{\frac{1}{2}\mathbb{E}_{lm_\mu}\mu \sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu} \sum_{k=lm_\mu}^{lm_\mu+m_\mu-1} \pi^2(\alpha_k^M)\,\mathrm{Tr}[f_{\alpha\alpha}(\alpha^M_{lm_\mu})w_k w_k']q_k^M\Big\},$$

is the same as that of

$$\begin{aligned}&\mathbb{E}g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)\Big\{\frac{1}{2}\sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu} \pi^2(\alpha^M_{lm_\mu})\,\mathrm{Tr}[f_{\alpha\alpha}(\alpha^M_{lm_\mu})]\delta_\mu q_M(\alpha_{lm_\mu})\Big\}\\ &\quad= \mathbb{E}g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)\Big\{\frac{1}{2}\sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu} \pi^2(\alpha^{\mu,M}(l\delta_\mu))\,\mathrm{Tr}[f_{\alpha\alpha}(\alpha^{\mu,M}(l\delta_\mu))]\delta_\mu q_M(\pi(\alpha^{\mu,M}(l\delta_\mu)))\Big\}.\end{aligned}$$

It then follows from weak convergence of $\alpha^{\mu,M}(\cdot)$ to $\alpha^M(\cdot)$ and the Skorohod representation,

$$\begin{aligned}&\mathbb{E}g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)\Big\{\frac{1}{2}\mathbb{E}_{lm_\mu}\mu \sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu} \sum_{k=lm_\mu}^{lm_\mu+m_\mu-1} \pi^2(\alpha_k^M)\,\mathrm{Tr}[f_{\alpha\alpha}(\alpha^M_{lm_\mu})w_k w_k']q_k^M\Big\}\\ &\quad\to \mathbb{E}g(\alpha^M(t_\imath) : \imath \le \kappa_1)\Big\{\frac{1}{2}\int_t^{t+s} \pi^2(\alpha^M(u))\,\mathrm{Tr}[f_{\alpha\alpha}(\alpha^M(u))]q_M(\alpha^M(u))du\Big\} \text{ as } \mu \to 0.\end{aligned} \tag{71}$$

Using the condition on the i.i.d. noise $\{w_k\}$, it is readily seen that

$$\begin{aligned}&\mathbb{E}g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)\Big\{\mathbb{E}_{lm_\mu}\sqrt{\mu} \sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu} f_\alpha'(\alpha^M_{lm_\mu}) \sum_{k=lm_\mu}^{lm_\mu+m_\mu-1} \pi(\alpha_k^M)w_k q_k^M\Big\}\\ &\quad= \mathbb{E}g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)\Big\{\sqrt{\mu} \sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu} f_\alpha'(\alpha^M_{lm_\mu}) \sum_{k=lm_\mu}^{lm_\mu+m_\mu-1} \mathbb{E}_{lm_\mu}\pi(\alpha_k^M)\mathbb{E}_{lm_\mu}w_k q_k^M\Big\}\\ &\quad\to 0 \text{ as } \mu \to 0.\end{aligned} \tag{72}$$

Next, using the continuity of $\pi(\cdot)$, $\pi_\alpha(\cdot)$, $f_\alpha(\cdot)$, together with the weak convergence of $\alpha^{\mu,M}(\cdot)$ to $\alpha^M(\cdot)$, the Skorohod representation, the notation q_k^M defined before, and the notation convention $q^M_{lm_\mu} = q_M(\alpha^M_{lm_\mu})$ and $q_l^M = q_M(\alpha^M(l\delta_\mu))$, we have

$$\begin{aligned}&\lim_{\mu\to 0}\mathbb{E}\Big[g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)\Big\{\mathbb{E}_{lm_\mu}\mu \sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu} f_\alpha'(\alpha^M_{lm_\mu}) \sum_{k=lm_\mu}^{lm_\mu+m_\mu-1} \pi_\alpha(\alpha_k^M)\pi(\alpha_k^M)q_k^M\Big\}\Big]\\ &\quad= \lim_{\mu\to 0}\mathbb{E}\Big[g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)\Big\{\mathbb{E}_{lm_\mu}\mu \sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu} f_\alpha'(\alpha^M_{lm_\mu}) \sum_{k=lm_\mu}^{lm_\mu+m_\mu-1} \pi_\alpha(\alpha^M_{lm_\mu})\pi(\alpha^M_{lm_\mu})q^M_{lm_\mu}\Big\}\Big]\\ &\quad= \lim_{\mu\to 0}\mathbb{E}\Big[g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)\Big\{\mathbb{E}_{lm_\mu} \sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu} f_\alpha'(\alpha^{\mu,M}(l\delta_\mu))\pi_\alpha(\alpha^{\mu,M}(l\delta_\mu))\pi(\alpha^{\mu,M}(l\delta_\mu))q_l^M\delta_\mu\Big\}\Big]\\ &\quad= \mathbb{E}\Big[g(\alpha^M(t_\imath) : \imath \le \kappa_1)\Big\{\int_t^{t+s} f_\alpha'(\alpha^M(u))\pi_\alpha(\alpha^M(u))\pi(\alpha^M(u))q_M(\alpha^M(u))du\Big\}\Big].\end{aligned} \tag{73}$$

Note that

$$\begin{aligned}
&\mathbb{E}\Big[g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)\Big\{\mathbb{E}_{lm_\mu}\sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu} f'_\alpha(\alpha^M_{lm_\mu})\sum_{k=lm_\mu}^{lm_\mu+m_\mu-1}\frac{\mu}{\Delta^N}K\Big(\frac{\theta_k-\alpha^M_k}{\Delta}\Big)\frac{\beta}{2}\widetilde{r}(\theta_k,\xi_k)\pi(\alpha^M_k)q^M_k\Big\}\Big]\\
&=\mathbb{E}\Big[g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)\Big\{\frac{\beta}{2}\sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu}\delta_\mu f'_\alpha(\alpha^M_{lm_\mu})\pi(\alpha^M_{lm_\mu})\\
&\qquad\times\frac{1}{\Delta^N}\frac{1}{m_\mu}\sum_{k=lm_\mu}^{lm_\mu+m_\mu-1}\mathbb{E}_{lm_\mu}K\Big(\frac{\theta_k-\alpha^M_k}{\Delta}\Big)\widetilde{r}(\theta_k,\xi_k)q^M_k\Big\}\Big]+o(1),
\end{aligned}\tag{74}$$

where $o(1)\to 0$ as $\mu\to 0$ uniformly in t. By the continuity of $\pi(\cdot)$ and $\widetilde{r}(\cdot,\xi)$ for each ξ,

$$\begin{aligned}
\psi_\mu &= \frac{1}{\Delta^N}\frac{1}{m_\mu}\sum_{k=lm_\mu}^{lm_\mu+m_\mu-1}\mathbb{E}_{lm_\mu}K\Big(\frac{\theta_k-\alpha^M_k}{\Delta}\Big)\widetilde{r}(\theta_k,\xi_k)q^M_k\\
&=\frac{1}{\Delta^N}\frac{1}{m_\mu}\sum_{k=lm_\mu}^{lm_\mu+m_\mu-1}\mathbb{E}_{lm_\mu}\Big[\int K\Big(\frac{\theta-\alpha^M_k}{\Delta}\Big)\widetilde{r}(\theta,\xi_k)\pi(\theta)d\theta\Big]_{\theta=\theta_k}q^M_k\\
&\quad+\frac{1}{\Delta^N}\frac{1}{m_\mu}\sum_{k=lm_\mu}^{lm_\mu+m_\mu-1}\mathbb{E}_{lm_\mu}\Big[\int K\Big(\frac{\theta-\alpha^M_k}{\Delta}\Big)\widetilde{r}(\theta,\xi_k)[\pi(\theta|\mathcal{F}_{lm_\mu})-\pi(\theta)]d\theta\Big]_{\theta=\theta_k}q^M_k
\end{aligned}$$

In view of (A2), the last term above contributes nothing to the limit. By virtue of (65),

$$\begin{aligned}
&\frac{1}{\Delta^N}\frac{1}{m_\mu}\sum_{k=lm_\mu}^{lm_\mu+m_\mu-1}\mathbb{E}_{lm_\mu}\Big[\int K\Big(\frac{\theta-\alpha^M_k}{\Delta}\Big)\widetilde{r}(\theta,\xi_k)\pi(\theta)d\theta\Big]_{\theta=\theta_k}q^M_k\\
&=\frac{1}{m_\mu}\sum_{k=lm_\mu}^{lm_\mu+m_\mu-1}\mathbb{E}_{lm_\mu}\widetilde{r}(\alpha^M_k,\xi_k)\pi(\alpha^M_k)q^M_k+o(1),
\end{aligned}$$

where $o(1)\to 0$ in probability. Thus we have

$$\begin{aligned}
\psi_\mu &= \frac{1}{m_\mu}\sum_{k=lm_\mu}^{lm_\mu+m_\mu-1}\mathbb{E}_{lm_\mu}\widetilde{r}(\alpha^M_k,\xi_k)\pi(\alpha^M_k)q^M_k+o(1)\\
&=\frac{1}{m_\mu}\sum_{k=lm_\mu}^{lm_\mu+m_\mu-1}\mathbb{E}_{lm_\mu}\widetilde{r}(\alpha^M_{lm_\mu},\xi_k)\pi(\alpha^M_{lm_\mu})q^M_k+o(1)\\
&=\frac{1}{m_\mu}\sum_{k=lm_\mu}^{lm_\mu+m_\mu-1}\mathbb{E}_{lm_\mu}\widetilde{r}(\alpha^{\mu,M}(l\delta_\mu),\xi_k)\pi(\alpha^{\mu,M}(l\delta_\mu))q^M_l+o(1),
\end{aligned}\tag{75}$$

where $o(1)\to 0$ in probability as $\mu\to 0$, because of the continuity of $\pi(\cdot)$ and $\widetilde{r}(\cdot,\xi)$ for each ξ. Letting $l\delta_\mu\to u$ as $\mu\to 0$, then for any $lm_\mu\le k\le m_\mu+m_\mu$, $\mu k\to u$. Using the weak convergence of $\alpha^{\mu,M}(\cdot)$ to $\alpha^M(\cdot)$ and the Skorohod representation, we can approximate $\widetilde{r}(\alpha^{\mu,M}(l\delta_\mu),\xi_k)\pi(\alpha^{\mu,M}(l\delta_\mu))q_M(\alpha^{\mu,M}(l\delta_\mu))$ by $\widetilde{r}(\alpha^M(u),\xi_k)\pi(\alpha^M(u))q_M(\alpha^M(u))$ with an error going to 0. Because $\alpha^M(\cdot)$ is bounded, for each $\gamma>0$, we can choose $\{O^\gamma_i : i\le i_\gamma\}$ as a finite collection of disjoint sets of diameter no larger than γ whose union covers the range of $\alpha^M(\cdot)$. Thus, $\alpha^M(\cdot)$ can be approximated by $\sum_{i=1}^{i_\gamma} \alpha_i^\gamma 1_{\{\alpha^M(u)\in O_i^\gamma\}}$. Consequently,

$$\psi_\mu = \frac{1}{m_\mu} \sum_{i=1}^{i_\gamma} \sum_{k=lm_\mu}^{lm_\mu+m_\mu-1} \mathbb{E}_{lm_\mu} \widetilde{r}(\alpha_i^\gamma, \xi_k)\pi(\alpha_i^\gamma) 1_{\{\alpha^M(u)\in O_i^\gamma\}} q_M(\alpha^M(u)) + o(1), \tag{76}$$

where $o(1) \to 0$ in probability. Now it is clear that condition (A3) and hence (61) can be used. Using (76) and (61) together with (74) and detailed calculation yields that

$$\mathbb{E}\Big[g(\alpha^{\mu,M}(t_\imath) : \imath \le \kappa_1)\Big\{\mathbb{E}_{lm_\mu} \sum_{l=t/\delta_\mu}^{(t+s)/\delta_\mu} f_\alpha'(\alpha_{lm_\mu}^M) \sum_{k=lm_\mu}^{lm_\mu+m_\mu-1} \frac{\mu}{\Delta^N} K\Big(\frac{\theta_k - \alpha_k^M}{\Delta}\Big) \frac{\beta}{2} \widetilde{r}(\theta_k, \xi_k)\pi(\alpha_k^M) q_k^M\Big\}\Big]$$
$$\to \mathbb{E}\Big[g(\alpha^M(t_\imath) : \imath \le \kappa_1)\Big\{\frac{\beta}{2}\int_t^{t+s} R_\alpha(\alpha^M(u))\pi^2(\alpha^M(u)) q_M(u)du\Big\}\Big]. \tag{77}$$

Using (68) and (69), and combining the estimates and calculation in (70)-(77) lead to (67). Therefore, we arrive at that $\alpha^M(\cdot)$ is the solution of the martingale problem with operator $\mathcal{L}^M$ given in (66).

Step 4. Let the truncation level $M \to \infty$. In the last step, we let $M \to \infty$ to obtain the convergence of the un-truncated process $\alpha^\mu(\cdot)$. The details are as in Kushner (1984, pp. 44-46). The verbatim argument is thus omitted.

Now, our arguments in Steps 1-4 yield the desired result Theorem 1. The proof of the theorem is concluded.

4.3 Comments

We make two remarks below.

- We proved Theorem 1 above for algorithm (56); equivalently (2). The proof of convergence of (57) can be carried out similarly. The main difference is that we are utilizing the kernel $K(\cdot)$ to incorporate θ_k used in the algorithm. There is no additional technical difficulty.
- Note that in a way, (56) can be considered to be more efficient than (57). First, because $\pi(\alpha)$ is available, (56) is more direct. Second, using $\pi(\alpha)$ and $\pi_\alpha(\alpha)$ in lieu of using $\pi(\theta)$ and $\pi_\alpha(\theta)$ together with the kernel $K(\cdot)$ avoids an additional averaging and the involvement of a Dirac δ-like function.

5. Tracking Analysis of IRL in Non-Stationary Environment

An important feature of the IRL algorithm (2) is its constant step size μ (as opposed to a decreasing step size). This facilities estimating (tracking) time evolving reward functions. This section analyzes the ability of IRL algorithm to track a time-varying reward function.

Since we are estimating a time evolving reward, we first give a model for the evolution of the reward $R(\theta)$ over time. Below, the Markov chain $\{x_k\}$ will be used as a *hyper-parameter* to model the evolution of the time varying reward, which we will denote as $R(\theta, x_k)$. By hyper-parameter we mean that the Markov chain model is not known or used by the IRL algorithm (2). The Markov chain assumption is used only in our convergence analysis to determine how well does the IRL algorithm estimates (tracks) the reward $R(\theta, x_k)$ that jump changes (evolves) according to an unknown Markov chain x_k.

We assume that the RL agents perform gradient algorithm (1) by evaluating the sequence of gradients $\{\nabla_\theta r_k(\theta_k, x_k)\}$. Note that both the RL and IRL do not know the sample path $\{x_k\}$. We will use similar notation to Sec.4:

- Denote $\nabla_\theta r_k(\theta_k, x_k)$ as $\widetilde{r}(\theta_k, \xi_k, x_k)$,
- We use $\pi_\alpha(\cdot)$ to denote $\nabla_\alpha \pi(\cdot)$.

5.1 Assumptions

We focus on the following algorithm

$$\alpha_{k+1} = \alpha_k + \frac{\mu}{\Delta^N} K\Big(\frac{\theta_k - \alpha_k}{\Delta}\Big) \frac{\beta}{2} \widetilde{r}(\theta_k, \xi_k, x_k)\pi(\alpha_k) + \mu \pi_\alpha(\alpha_k)\pi(\alpha_k) + \sqrt{\mu}\pi(\alpha_k) w_k, \quad (78)$$

The main assumptions are as follows.

(M1) (Markovian hyper-parameter) Let $\{x_k, k \geq 0\}$ be a Markov chain with finite state space $\mathcal{X} = \{1, \ldots, X\}$ and transition probability matrix $I + \eta Q$, where $\eta > 0$ is a small parameter and $Q = (q_{ij})$ is an $X \times X$ irreducible generator (matrix) (Yin and Zhang, 2013, p.23) with

$$q_{ij} \geq 0, \quad i \neq j, \qquad \sum_j q_{ij} = 0, \quad i \in \mathcal{X},$$

also $\{x_k\}$ is independent of $\{\theta_k\}$ and $\{w_k\}$.

(M2) Assumption (A1) holds on $\widetilde{r}(\cdot, \xi, i)$ for each fixed state $i \in \mathcal{X}$. Also (A2), (A3), (A4) hold.

5.2 Main Result

Recall that μ is the step size of the IRL algorithm while η reflects the rate at which the hyper-parameter Markov chain x_k evolves. In the following tracking analysis of IRL algorithm (2) , we will consider three cases, $\mu = O(\eta)$, $\mu \ll \eta$, and $\mu \gg \eta$. The three cases represent three different types of asymptotic behavior. If $\mu \gg \eta$, the frequency of changes of the Markov chain is very slow. Thus, we are treating a case similar to a constant parameter, or we essentially deal with a "single" objective function. If $\mu \ll \eta$, then the Markov chain jump changes frequently. So what we are optimizing is a function $\sum_{i=1}^X R(\alpha, i)\nu_i$, where ν_i is the stationary distribution associated with the generator Q. If $\mu = O(\eta)$, then the Markov chain changes in line with the optimization recursion. In this case, we obtain switching limit Langevin diffusion.

In Theorem 2 below, for brevity we use $\mu = \eta$ for $\mu = O(\eta)$, $\eta = \mu^{1+\widetilde{\Delta}}$ for $\eta = o(\mu)$ and $\eta = \mu^{\widetilde{\Delta}}$ for $\mu = o(\eta)$, respectively. These cover all three possible cases of the rate at which the hyper-parameter evolves compared to the dynamics of the Langevin IRL algorithm.

Theorem 2 *Consider the IRL algorithm* (78). *Under Assumptions* (M1) *and* (M2)*, assuming that* (79)*, or* (80)*, or* (81) *has a unique solution in the sense in distribution. Then the following results hold.*

1. *Assume $\mu = \eta$. Then as $\mu \downarrow 0$, the interpolated process $(\alpha^\mu(\cdot), x^\mu(\cdot))$ converges weakly to the switching diffusion $(\alpha(\cdot), x(\cdot))$ satisfying*

$$d\alpha(t) = \Big[\frac{\beta}{2}\pi^2(\alpha(t))R_\alpha(\alpha(t), x(t)) + \pi_\alpha(\alpha(t))\pi(\alpha(t))\Big]dt + \pi(\alpha(t))dW(t), \tag{79}$$

where $W(\cdot)$ is a standard Brownian motion with mean 0 and covariance being the identity matrix $I \in \mathbb{R}^{N\times N}$, and $x(\cdot)$ is a continuous-time Markov chain with generator Q.

2. *Suppose $\eta = \mu^{1+\widetilde{\Delta}}$ with $\widetilde{\Delta} > 0$ and denote the initial distribution of $x^\eta(0)$ by p_ι (independent of η) for each $\iota \in \mathcal{X}$. Then as $\mu \downarrow 0$, the interpolated process $(\alpha^\mu(\cdot))$ converges weakly to the following diffusion process*

$$d\alpha(t) = \Big[\frac{\beta}{2}\pi^2(\alpha(t))\sum_{\iota\in\mathcal{X}} R_\alpha(\alpha(t), \iota)\, p_\iota + \pi_\alpha(\alpha(t))\pi(\alpha(t))\Big]dt + \pi(\alpha(t))dW(t), \tag{80}$$

3. *Suppose that $\eta = \mu^{\widetilde{\Delta}}$ with $0 < \widetilde{\Delta} < 1$ and denote the stationary distribution associated with the continuous-time Markov chain with generator Q by $\nu = (\nu_1, \dots, \nu_X)$. Then as $\mu \downarrow 0$, the interpolated process $(\alpha^\mu(\cdot))$ converges weakly to the following diffusion process*

$$d\alpha(t) = \Big[\frac{\beta}{2}\pi^2(\alpha(t))\sum_{\iota\in\mathcal{X}} R_\alpha(\alpha(t), \iota)\, \nu_\iota + \pi_\alpha(\alpha(t))\pi(\alpha(t))\Big]dt + \pi(\alpha(t))dW(t). \tag{81}$$

Remark. Theorem 2 presented the asymptotic behavior of the IRL algorithm (78) with Markovian switching. In accordance with the rates of variations of the adaptation rates (represented by the stepsize μ) and the switching rate (represented by the stepsize η), three cases are considered. Case 1 indicates that when μ is in line with η, the limit differential equation is a switching diffusion. Case 2 concentrates on the case that the switching is much slower than the stochastic approximation generated by the recursion. Thus, the limit Langevin equation is one in which the drift and diffusion coefficients are averaged out with respect to the initial distribution of the limit Markov chain. Roughly, it reveals that the "jump change" parameter $x(t)$ is more or less as a constant in the sense the coefficients are averages w.r.t. the initial distribution. Case 3 is the one that the Markov chain is changing much faster than the stochastic approximation rate. As a result, the "jump change" behavior is replaced by an average with respect to the stationary distribution of the Markov chain. Then we derive the associated limit Langevin equation. Again, the limit has no switching in it.

5.3 Proof of Theorem 2

We will prove Statement 1 for the case $\mu = \eta$. Consider $(\alpha^\mu(\cdot), x^\mu(\cdot))$, the pair of interpolated processes. We shall show that this pair of processes converges weakly to $(\alpha(\cdot), x(\cdot))$ such that the limit is a solution of (79) or equivalently, $(\alpha(\cdot), x(\cdot))$ is a solution of the martingale problem with an operator redefined by

$$\mathcal{L}f(\alpha, i) = f'_\alpha(\alpha, i)\Big[\frac{\beta}{2}\pi^2(\alpha, i)R_\alpha(\alpha) + \pi_\alpha(\alpha)\pi(\alpha)\Big] + \frac{1}{2}\pi^2(\alpha)\,\mathrm{Tr}[f_{\alpha\alpha}(\alpha, i)] + Qf(\alpha, \cdot)(i), \tag{82}$$

where

$$Qf(\alpha, \cdot)(i) = \sum_{j\in\mathcal{X}} q_{ij} f(\alpha, j), \quad \text{for each } \ i \in \mathcal{X}.$$

We still need to use an M truncation device (truncation on α). However, for notation simplicity, we suppress the M truncation. From (78), it is easily seen that

$$\alpha_{k+1} = \alpha_k + \frac{\mu}{\Delta^N} K\Big(\frac{\theta_k - \alpha_k}{\Delta}\Big)\frac{\beta}{2}\sum_{i\in\mathcal{X}} \widetilde{r}(\theta_k, \xi_k, i)\pi(\alpha_k)1_{\{x_k=i\}} + \mu\pi_\alpha(\alpha_k)\pi(\alpha_k) + \sqrt{\mu}\pi(\alpha_k)w_k. \tag{83}$$

To prove the tightness of $(\alpha^\mu(\cdot), x^\mu(\cdot))$, we prove the tightness $\{x^\mu(\cdot)\}$ first. This can be done by considering $\chi_k = (1_{\{x_k=1\}}, \dots, 1_{\{x_k=X\}}) \in \mathbb{R}^{1\times X}$, and defining $\chi^\mu(t) = \chi_k$ for $t \in [\mu k, \mu k + \mu)$. Denote by $\mathcal{F}^\mu_t$, the σ-algebra generated by $\{\xi_k, \theta_k, x_k, \alpha_0 : k \le t/\mu\}$, and denote the corresponding conditional expectation by $\mathbb{E}^\mu_t$. Because χ_k is a Markov chain and because of the independence of x_k with ξ_k and θ_k, we can show for any $\delta > 0$, $t > 0$, $s > 0$ with $s \le \delta$, for some random variable $\widehat{\gamma}^\mu(\delta) > 0$,

$$\sup_{0\le s\le\delta} \mathbb{E}^\mu_t[|\chi^\mu(t+s) - \chi^\mu(t)|^2 \big| \mathcal{F}^\mu_t] \le \mathbb{E}^\mu_t \widehat{\gamma}^\mu(\delta).$$

Furthermore,

$$\lim_{\delta\to 0}\limsup_{\mu\to 0} \mathbb{E}\widehat{\gamma}^\mu(\delta) = 0,$$

which implies the tightness of $\{\chi^\mu(\cdot)\}$ (see (Kushner, 1984, p. 47, Theorem 3) and hence the tightness of $\{x^\mu(\cdot)\}$. We can also prove the tightness of $\{\alpha^\mu(\cdot)\}$. Then the tightness of $\{\alpha^\mu(\cdot), x^\mu(\cdot)\}$ can be proved. The rest of the averaging procedure is similar to that of the proof of Theorem 1.

For the proofs of Statement 2, the case $\eta = \mu^{1+\widetilde{\Delta}}$, and Statement 3, the case $\eta = \mu^{\widetilde{\Delta}}$, the arguments are similar to Yin et al. (2013) Section 4.1 and Section 4.2, respectively. We thus omit the details.

6. Proof of Convergence of IRL Algorithm (20)

Here we prove weak convergence of the multi-kernel variance reduction IRL algorithm (20). The proof involves a novel application of the Bernstein von-Mises theorem (which in simple terms is a central limit theorem for a Bayesian estimator); see (88) below.

Recall that we write $\nabla r_k(\theta)$ as $\widetilde{r}(\theta, \xi_k)$ as in the proof of Theorem 1. The algorithm (20) is

$$\alpha_{k+1} = \alpha_k + \mu\,\frac{\beta}{2}\sum_{i=1}^{L_\mu} \frac{p(\alpha_k|\theta_i)\widetilde{r}(\theta_i, \xi_k)}{\sum_{l=1}^{L_\mu} p(\alpha_k|\theta_l)} + \sqrt{\mu} w_k, \tag{84}$$

where L_μ is so chosen that $L_\mu \to \infty$ as $\mu \to 0$.

We start with the following assumptions:

(B1) (A1) holds and the reward $R(\cdot)$ has continuous partial derivatives up to the second order and the second-order partial of R is uniformly bounded.

(B2) The $\{\theta_l\}$ is a stationary sequence $\theta_l \sim \pi(\cdot)$; $\{\theta_l\}$ is independent of $\{\xi_k\}$ and $\{w_k\}$, where $\{\xi_k\}$ and $\{w_k\}$ satisfy (A3).

(B3) For each fixed α, and each $i = 1, \dots, L_\mu$, define

$$\gamma_i(\alpha) = \frac{p(\alpha|\theta_i)}{\sum_{l=1}^{L_\mu} p(\alpha|\theta_l)}.$$

For each ξ and each α, as $\mu \to 0$, $L_\mu \to \infty$ and

$$\sum_{i=1}^{L_\mu} \gamma_i(\alpha)\widetilde{r}(\theta_i, \xi) \to \mathbb{E}\widetilde{r}(\theta, \xi|\alpha) \text{ w.p.1.} \tag{85}$$

(B4) $\mathbb{E}|\widetilde{r}(\theta_i, \xi_k)|^2 < \infty$ for each i and each k, and $\int(1 + |\nabla R(\theta)|^2)(p(\theta/\alpha)/\pi(\theta))^2\pi(\theta)d\theta < \infty$.

(B5) (a) The conditional probability density function

$$p(\alpha|\theta) = p_v(\theta - \alpha) \tag{86}$$

where $p_v(\cdot)$ is a symmetric density with zero mean and covariance $O(\Delta^2)I$. where I denotes the identity matrix.
(b) The Fisher information matrix $I_\theta = \int_{\mathbb{R}^N} \nabla \log p(\alpha|\theta)\, p(\alpha|\theta)\, d\alpha$ is invertible for all $\theta \in \mathbb{R}^N$.

Remarks. We briefly comment on the assumptions. (B1) is a smoothness assumption on $\widetilde{r}(\cdot, \xi)$ and $R(\cdot)$. The second order differentiability of $R(\cdot)$ is used in a Taylor series expansion in Proposition 4 to obtain the final stochastic diffusion limit. Note that (B1) is a stronger assumption than (A1).

Condition (B2) specifies the distribution of θ_l. We also assume that this sequence is independent of the ξ_k and w_k. The assumption builds on (A3).

(B3) is an averaging condition; i.i.d. samples $\{\theta_i\}$ is a special case. In fact, we only need the convergence to be in the sense of convergence in probability.

(B4) is a classical square integrability assumption for asymptotic normality.

Finally, (B5) is used in the Bernstein von-Mises theorem to show that the posterior $p(\theta|\alpha)$ is asymptotically normal and behaves as a Dirac delta as $\Delta \downarrow 0$; see Proposition .4 below.

As in our previous proofs, we define the interpolated process $\alpha^\mu(t) = \alpha_k$, $t \in [\mu k, \mu k + \mu)$. For convenience, the proof proceeds in two steps: In the first step, Proposition 3 below shows that $\alpha^\mu(\cdot)$ converges weakly to $\alpha(\cdot)$ such that $\alpha(t)$ satisfies the stochastic differential equation (24) w.r.t. conditional expectation $p(\theta|\alpha(t))$.

Proposition 3 *Assume conditions* (B1)–(B5) *hold and that the stochastic differential equation*

$$d\alpha(t) = \int_{\mathbb{R}^N} \frac{\beta}{2} \nabla R(\theta)\, p\big(\theta|\alpha(t)\big)\, d\theta\, dt + dW(t), \qquad \alpha(0) = \alpha_0 \tag{87}$$

has a unique solution in the sense in distribution for each initial condition. Then the interpolated process $\alpha^\mu(\cdot)$ converges weakly to $\alpha(\cdot)$ such that $\alpha(\cdot)$ is the solution of (87).

Note that in the above, we used the uniqueness solution in the weak or distribution sense. Such a uniqueness is equivalent to the uniqueness of the associated martingale problem; see Ethier and Kurtz (1986, p. 182) or Karatzas and Shreve (1991).

In the second step, we use the Bernstein von-Mises theorem below to characterize the posterior as a normal distribution when the parameter Δ in the likelihood density goes to zero. The Bernstein-von Mises theorem (Van der Vaart, 2000) implies that for small parameter Δ in the likelihood (21), the posterior converges to the Gaussian density $\mathbf{N}(\theta; \alpha, \Delta^2 I_{\bar\theta})$. More precisely,

$$\int |p(\theta|\alpha) - \mathbf{N}(\theta; \alpha, \Delta^2 I_{\bar\theta})|d\theta \to 0 \text{ in probability under } P_{\bar\theta} \text{ as } as\Delta \to 0. \tag{88}$$

Here $I_{\bar\theta} = \int_{\mathbb{R}^N} \nabla \log p(\alpha|\theta)\, p(\alpha|\theta)\, d\alpha|_{\theta=\bar\theta}$ is the Fisher information matrix evaluated at the parameter value[14] $\bar\theta$ and

$$\mathbf{N}(\theta; \alpha, \Delta^2 I_{\bar\theta}) = 2\pi^{-N/2} \exp\Big[-\frac{1}{2}(\theta-\alpha)'|\Delta^2 I_{\bar\theta}^{-1}|^{-1}(\theta-\alpha)\Big]. \tag{89}$$

In view of the parametrization by Δ above, $\alpha^\mu(\cdot)$ should be written as $\alpha^{\mu,\Delta}(\cdot)$.

Proposition 4 *Assume conditions* (B1) *to* (B5)*, and* (88) *hold. Then the limit in Proposition* 3 *can be written as* $\alpha^\Delta(t)$*. As* $\Delta \to 0$*,* $\alpha^\Delta(t)$ *has the limit* $\alpha(t)$ *satisfying*

$$d\alpha(t) = \frac{\beta}{2}\nabla R(\alpha(t))\, dt + dW(t), \qquad \alpha(0) = \alpha_0. \tag{90}$$

6.1 Proof Outline of Proposition 3

We present the main ideas of the proof and the underlying intuition. Define $\alpha^\mu(t) = \alpha_k$, for $t \in [\mu k, \mu k + \mu)$. As in the proof of Theorem 1 in Sec.4.2, we should still use a truncation device and use the martingale problem formulation. However, to present the main idea without overburdening with technical details, we will use simpler and intuitive ideas. Thus we simply assume that the iterates are bounded. For example, we should use a smooth function with compact support $f(\cdot)$ as in the proof of Theorem 1. However, for simplicity of argument, we will illustrate the idea without using this function $f(\cdot)$; we will also suppress the truncation notation.

Denote by $\mathcal{F}^\mu_t$, the σ-algebra generated by $\{\theta_k, \xi_k, \alpha_0 : k \le \lfloor t/\mu \rfloor\}$, where $\lfloor s \rfloor$ denotes the integer part of s. In what follows, we shall suppress the floor function notation. Denote by $\mathbb{E}^\mu_t$, the conditional expectation with respect to $\mathcal{F}^\mu_t$. We also use $\mathbb{E}_{\xi_k}$ to denote the conditioning on $\{\xi_j : j \le k\}$. For any $\delta > 0$, $t > 0$, $s > 0$ and $s \le \delta$, by the boundedness of the iterates, conditions (B1), (B2), the form of $\gamma_i(\alpha)$ in (B3), and (B4), we have

$$\begin{aligned}
&\mathbb{E}^\mu_t|\alpha^\mu(t+s) - \alpha^\mu(t)|^2 \\
&\le K\Big[\mathbb{E}^\mu_t\Big|\mu \sum_{k=t/\mu}^{(t+s)/\mu-1} \sum_{i=1}^{L_\mu} \gamma_i(\alpha_k)\widetilde r(\theta_i,\xi_k)\Big|^2 + \mathbb{E}^\mu_t\Big|\sqrt{\mu}\sum_{k=t/\mu}^{(t+s)/\mu-1} w_k\Big|^2\Big] \\
&\le Ks\mu \sum_{k=t/\mu}^{(t+s)/\mu-1} \mathbb{E}^\mu_t\Big|\sum_{i=1}^{L_\mu}\gamma_i(\alpha_k)\widetilde r(\theta_i,\xi_k)\Big|^2 + K\mu\mathbb{E}^\mu_t \sum_{k=t/\mu}^{(t+s)/\mu-1} w_k' w_k \le \mathbb{E}^\mu_t \widehat\gamma^\mu(\delta),
\end{aligned}$$

where $\widehat\gamma^\mu(\delta)$ is a random variable. Moreover,

$$\lim_{\delta\to 0}\limsup_{\mu\to 0} \mathbb{E}\widehat\gamma^\mu(\delta) = 0.$$

Thus the tightness of $\{\alpha^\mu(\cdot)\}$ is obtained; see Kushner (1984, p. 47).

By Prohorov's theorem, we can extract a weakly convergent subsequence. Select such a sequence and still use μ as its index (for notional simplicity) with limit $\alpha(\cdot)$. By Skorohod representation (without changing notation), $\alpha^\mu(\cdot)$ converges w.p.1 to $\alpha(\cdot)$. Now for any $t > 0$ and $s > 0$,

$$\alpha^\mu(t+s) - \alpha^\mu(t) = \frac{\beta}{2}\mu \sum_{k=t/\mu}^{(t+s)/\mu-1} \sum_{i=1}^{L_\mu} \gamma_i(\alpha_k)\widetilde r(\theta_i,\xi_k) + \sqrt{\mu}\sum_{k=t/\mu}^{(t+s)/\mu-1} w_k. \tag{91}$$

14. It suffices to choose any $\bar\theta$ such that $\alpha \sim p(\cdot|\bar\theta)$. The precise value of $\bar\theta$ need not be known and is irrelevant to our analysis.

Define

$$W^\mu(t) = \sqrt{\mu} \sum_{k=0}^{(t/\mu)-1} w_k.$$

By using the classical functional invariance theorem, clearly $W^\mu(\cdot)$ converges weakly to $W(\cdot)$ a standard Brownian motion. As a consequence,

$$\begin{aligned} W^\mu(t+s) - W^\mu(t) &= \sqrt{\mu} \sum_{k=t/\mu}^{(t+s)/\mu-1} w_k \\ &\to W(t+s) - W(t) \end{aligned}$$

by the weak convergence and the Skorohod representation. To determine the limit of the drift term, we proceed similarly to the proof of Theorem 1. In view of (21), $\gamma_i(\alpha)$ is continuous (and in fact smooth) w.r.t. α. We use the finite value approximation argument as just above (76) together with the averaging condition in (B3). That is, for each $\tilde{\eta} > 0$, we can choose $\{O_j^{\tilde{\eta}} : j \le j_{\tilde{\eta}}\}$ as a finite collection of disjoint sets of diameter no larger than $\tilde{\eta}$ whose union covers the range of $\alpha^\mu(u)$ so $\alpha^\mu(u)$ can be approximated by $\sum_{j=1}^{j_{\tilde{\eta}}} \alpha_j^{\tilde{\eta}} 1_{\{\alpha^{\tilde{\eta}}(u) \in O_j^{\tilde{\eta}}\}}$. Thus using the notation as in the proof of Theorem 1, and choosing any positive integer κ_1 and $t_\iota \le t$ with $\iota \le \kappa_1$,

$$\begin{aligned} &\lim_{\mu\to 0} \mathbb{E} g(\alpha^\mu(t_\iota) : \iota \le \kappa_1) \Big[\mu \sum_{k=t/\mu}^{(t+s)/\mu-1} \sum_{i=1}^{L_\mu} \gamma_i(\alpha_k) \widetilde{r}(\theta_i, \xi_k) \Big] \\ &= \lim_{\mu\to 0} \mathbb{E} g(\alpha^\mu(t_\iota) : \iota \le \kappa_1) \Big[\sum_{l\delta_\mu = t}^{t+s} \delta_\mu \frac{1}{m_\mu} \sum_{k=lm_\mu}^{lm_\mu+m_\mu-1} \sum_{i=1}^{L_\mu} \gamma_i(\alpha_k) \widetilde{r}(\theta_i, \xi_k) \Big] \\ &= \lim_{\mu\to 0} \mathbb{E} g(\alpha^\mu(t_\iota) : \iota \le \kappa_1) \Big[\sum_{l\delta_\mu = t}^{t+s} \delta_\mu \frac{1}{m_\mu} \sum_{k=lm_\mu}^{lm_\mu+m_\mu-1} \mathbb{E}_{lm_\mu} \sum_{i=1}^{L_\mu} \gamma_i(\alpha_{lm_\mu}) \widetilde{r}(\theta_i, \xi_k) \Big] \\ &= \lim_{\mu\to 0} \mathbb{E} g(\alpha^\mu(t_\iota) : \iota \le \kappa_1) \Big[\sum_{l\delta_\mu = t}^{t+s} \delta_\mu \frac{1}{m_\mu} \sum_{k=lm_\mu}^{lm_\mu+m_\mu-1} \mathbb{E}_{lm_\mu} \sum_{j=1}^{j_{\tilde{\eta}}} \sum_{i=1}^{L_\mu} \gamma_i(\alpha_j^{\tilde{\eta}}) \widetilde{r}(\theta_i, \xi_k) 1_{\{\alpha^{\tilde{\eta}}(u) \in O_j^{\tilde{\eta}}\}} \Big] \\ &= \lim_{\mu\to 0} \mathbb{E} g(\alpha^\mu(t_\iota) : \iota \le \kappa_1) \Big[\sum_{l\delta_\mu = t}^{t+s} \delta_\mu \frac{1}{m_\mu} \sum_{k=lm_\mu}^{lm_\mu+m_\mu-1} \mathbb{E}_{lm_\mu} \sum_{j=1}^{j_{\tilde{\eta}}} \mathbb{E}_{\xi_k} [\widetilde{r}(\theta, \xi_k) | \alpha_j^{\tilde{\eta}}] 1_{\{\alpha^{\tilde{\eta}}(u) \in O_j^{\tilde{\eta}}\}} \Big] \\ &= \mathbb{E} g(\alpha(t_\iota) : \iota \le \kappa_1) \Big[\int_t^{t+s} \int_{\mathbb{R}^N} \nabla R_\theta(\theta) p(\theta | \alpha(u)) d\theta du \Big], \end{aligned} \tag{92}$$

where $\mathbb{E}_{\xi_k}$ denotes the conditioning on $\{\xi_j : j \le k\}$. In the above, we used (85), and noted that letting $\mu l m_\mu \to u$ yields $\mu k \to u$ for $lm_\mu \le k \le lm_\mu + m_\mu$. We also used (B5). Putting the estimates together, we obtain the desired limit.

6.2 Proof Outline of Proposition 4

To prove Proposition 4, using (B5), by virtue of (88), $p(\theta|\alpha)$ can be approximated by $\mathbf{N}(\theta; \alpha, \Delta^2 I_{\bar{\theta}})$, the normal density given by (89). For notational convenience denote $\widetilde{p}(\theta, \alpha) \overset{\triangle}{=} \mathbf{N}(\theta; \alpha, \Delta^2 I_{\bar{\theta}})$ below. Now, we work with $\Delta \to 0$. By Taylor expansion,

$$\nabla R(\theta) = \nabla R(\alpha) + \nabla^2 R(\alpha_+)[\theta - \alpha],$$

where $\nabla^2 R$ is the Hessian (the second partial derivatives) of R, and α_+ is on the line segment joining θ and α. Choose v so that $\varepsilon l m_\varepsilon \to v$. As a result, for any k satisfying $l m_\varepsilon \leq k \leq l m_\varepsilon + m_\varepsilon$, $\varepsilon k \to v$. It follows that

$$\begin{aligned}
\int_{\mathbb{R}^N} \nabla R(\theta) p(\theta|\alpha(v)) d\theta &= \int_{\mathbb{R}^N} \nabla R(\theta) \widetilde{p}(\theta, \alpha(v)) d\theta + o_\Delta(1) \\
&= \int_{\mathbb{R}^N} \nabla R(\alpha) \widetilde{p}(\theta, \alpha(v)) d\theta \\
&\quad + \int_{\mathbb{R}^N} \nabla^2 R(\alpha_+(v))[\theta - \alpha(v)] \widetilde{p}(\theta, \alpha(v)) d\theta + o_\Delta(1) \\
&= \nabla R(\alpha(v)) + o_\Delta(1) \\
&\to \nabla R(\alpha(v)) \text{ as } \Delta \to 0,
\end{aligned} \tag{93}$$

where $o_\Delta(1) \to 0$ in probability as $\Delta \to 0$. Note that in the above, the form of the density implies

$$\int_{\mathbb{R}^N} \nabla^2 R(\alpha_+(v))[\theta - \alpha(v)] \widetilde{p}(\theta, \alpha(v)) d\theta = 0.$$

7. Conclusions and Extensions

This paper has presented and analyzed the convergence of passive Langevin dynamics algorithms for adaptive inverse reinforcement learning (IRL). Given noisy gradient estimates of a possibly time evolving reward function R, the Langevin dynamics algorithm generates samples $\{\alpha_k\}$ from the Gibbs measure $p(\alpha) \propto \exp\big(\beta R(\alpha)\big)$; so the log of the empirical distribution of $\{\alpha_k\}$ serves as a non-parametric estimator for $R(\alpha)$. The proposed algorithm is a *passive* learning algorithm since the gradients are not evaluated at α_k by the inverse learner; instead the gradients are evaluated at the random points θ_k chosen by the gradient (RL) algorithm. This passive framework is natural in an IRL where the inverse learner passively observes forward learners.

Apart from the main IRL algorithm (2), we presented a two-time scale IRL algorithm for variance reduction, an active IRL algorithm which deals with mis-specified gradients, and a non-reversible diffusion IRL algorithm with larger spectral gap and therefore faster convergence to the stationary distribution. We presented three detailed numerical examples: inverse Bayesian learning, a large dimensional IRL problem in logistic learning involving a real dataset, and IRL for a constrained Markov decision process. Finally, we presented a complete weak convergence proof of the IRL algorithm using martingale averaging methods. We also analyzed the tracking capabilities of the IRL algorithm when the utility function jump changes according to a slow (but unknown) Markov chain.

Extensions. A detailed proof of the two-time scale variance reduction algorithm involves Bayesian asymptotics, namely, the Bernstein von Mises theorem. Since the submission of the current paper, in a recent work (Krishnamurthy and Yin, 2020), we have developed a complete convergence proof. It is important to note that the IRL algorithms proposed in this paper are adaptive: given the estimates from an adaptive gradient algorithm, the IRL algorithm learns the utility function. In other words, we have a gradient algorithm operating in series with a Langevin dynamics algorithm. In future work it is of interest to study the convergence properties of multiple such cascaded Langevin dynamics and gradient algorithms.

The recent paper by Kamalaruban et al. (2020) shows that classical Langevin dynamics yields more robust RL algorithms compared to classic stochastic gradient. In analogy to Kamalaruban et al.

(2020), in future work it is worthwhile exploring how our passive Langevin dynamics framework can be viewed as a robust version of classical passive stochastic gradient algorithms.

Finally, this paper analyzed the weak convergence and tracking properties of passive Langevin dynamic algorithms. In future work it is of interest to analyze the asymptotic convergence rate and spectral gap of the diffusion process induced by the proposed algorithm. This will also facilitate quantifying how the convergence rate is affected when the step size ε_n of each RL agent n is chosen randomly (and unknown to the inverse learner).

Acknowledgment

This research was funded in part by the U.S. Army Research Office under grants W911NF-21-1-0093 and W911NF-19-1-0365, and in part by the Air Force Office of Scientific Research under grant FA9550-18-1-0268.

References

F. V. Abad and V. Krishnamurthy. Constrained stochastic approximation algorithms for adaptive control of constrained Markov decision processes. In *42nd IEEE Conference on Decision and Control*, pages 2823–2828, 2003.

P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1, 2004.

S. Afriat. The construction of utility functions from expenditure data. *International economic review*, 8(1):67–77, 1967.

E. Altman. *Constrained Markov Decision Processes*. Chapman and Hall, London, 1999.

A. R. Barron. The strong ergodic theorem for densities: generalized Shannon-McMillan-Breiman theorem. *The Annals of Probability*, 13(4):1292–1303, 1985.

A. Benveniste, M. Metivier, and P. Priouret. *Adaptive Algorithms and Stochastic Approximations*, volume 22 of *Applications of Mathematics*. Springer-Verlag, 1990.

D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, Belmont, MA., 1996.

P. Billingsley. *Convergence of Probability Measures*. Wiley, New York, second edition, 1999.

V. Borkar and R. Jain. Risk-constrained markov decision processes. In *49th IEEE Conference on Decision and Control (CDC)*, pages 2664–2669. IEEE, 2010.

O. Cappe, E. Moulines, and T. Ryden. *Inference in Hidden Markov Models*. Springer-Verlag, 2005.

W. Diewert. Afriat's theorem and some extensions to choice under uncertainty. *The Economic Journal*, 122(560):305–331, 2012.

D. Djonin and V. Krishnamurthy. Q-learning algorithms for constrained Markov decision processes with randomized monotone policies: Applications in transmission control. *IEEE Transactions on Signal Processing*, 55(5):2170–2181, 2007.

S. N. Ethier and T. G. Kurtz. *Markov Processes—Characterization and Convergence*. Wiley, 1986.

S. B. Gelfand and S. K. Mitter. Recursive stochastic algorithms for global optimization in Rˆd. *SIAM Journal on Control and Optimization*, 29(5):999–1018, 1991.

P. Guan, M. Raginsky, and R. M. Willett. Online markov decision processes with Kullback–Leibler control cost. *IEEE Transactions on Automatic Control*, 59(6):1423–1438, 2014.

W. Hardle and R. Nixdorf. Nonparametric sequential estimation of zeros and extrema of regression functions. *IEEE transactions on information theory*, 33(3):367–372, 1987.

J. Ho and S. Ermon. Generative adversarial imitation learning. In *Advances in neural information processing systems*, pages 4565–4573, 2016.

C.-R. Hwang, S.-Y. Hwang-Ma, and S.-J. Sheu. Accelerating gaussian diffusions. *The Annals of Applied Probability*, pages 897–913, 1993.

C.-R. Hwang, S.-Y. Hwang-Ma, S.-J. Sheu, et al. Accelerating diffusions. *The Annals of Applied Probability*, 15(2):1433–1444, 2005.

P. Kamalaruban, Y.-T. Huang, Y.-P. Hsieh, P. Rolland, C. Shi, and V. Cevher. Robust reinforcement learning via adversarial training with langevin dynamics. *arXiv preprint arXiv:2002.06063*, 2020.

I. Karatzas and S. Shreve. *Brownian Motion and Stochastic Calculus*. Springer, second edition, 1991.

V. Y. Katkovnik. Linear estimates and stochastic optimization problems. *Nauka, Moscow*, 1976.

V. Krishnamurthy. *Partially Observed Markov Decision Processes. From Filtering to Controlled Sensing*. Cambridge University Press, 2016.

V. Krishnamurthy and F. Vazquez Abad. Real-time reinforcement learning of constrained Markov decision processes with weak derivatives. *arXiv preprint arXiv:1110.4946*, 2018.

V. Krishnamurthy and G. Yin. Multi-kernel passive stochastic gradient algorithms. *arXiv preprint arXiv:2008.10020*, 2020.

H. J. Kushner. *Approximation and Weak Convergence Methods for Random Processes, with applications to Stochastic Systems Theory*. MIT Press, Cambridge, MA, 1984.

H. J. Kushner and G. Yin. *Stochastic Approximation Algorithms and Recursive Algorithms and Applications*. Springer-Verlag, 2nd edition, 2003.

L. Ljung. Analysis of recursive stochastic algorithms. *IEEE Transactions on Auto. Control*, AC-22 (4):551–575, 1977.

A. V. Nazin, B. T. Polyak, and A. B. Tsybakov. Passive stochastic approximation. *Automat. Remote Control*, (50):1563–1569, 1989.

A. Nedic and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, 2009.

A. Ng and S. Russell. Algorithms for inverse reinforcement learning. In *Proc. 17th International Conf. Machine Learning*, pages 663–670, 2000.

M. H. Ngo and V. Krishnamurthy. Monotonicity of constrained optimal transmission policies in correlated fading channels with ARQ. *IEEE Transactions on Signal Processing*, 58(1):438–451, 2010.

T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters. An algorithmic perspective on imitation learning. *arXiv preprint arXiv:1811.06711*, 2018.

G. A. Pavliotis. *Stochastic processes and applications: diffusion processes, the Fokker-Planck and Langevin equations*, volume 60. Springer, 2014.

G. Pflug. *Optimization of Stochastic Models: The Interface between Simulation and Optimization*. Kluwer Academic Publishers, 1996.

M. Puterman. *Markov Decision Processes*. John Wiley, 1994.

M. Raginsky, A. Rakhlin, and M. Telgarsky. Non-convex learning via stochastic gradient Langevin dynamics: a nonasymptotic analysis. *arXiv preprint arXiv:1702.03849*, 2017.

P. Révész. How to apply the method of stochastic approximation in the non-parametric estimation of a regression function. *Statistics: A Journal of Theoretical and Applied Statistics*, 8(1):119–126, 1977.

C. P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer-Verlag, 2013.

S. Ross. *Introduction to Stochastic Dynamic Programming*. Academic Press, San Diego, California., 1983.

S. Ross. *Simulation*. Academic Press, 5 edition, 2013.

J. Spall. *Introduction to Stochastic Search and Optimization*. Wiley, 2003.

O. Stramer and R. L. Tweedie. Langevin-type models I: Diffusions with given stationary distributions and their discretizations. *Methodology and Computing in Applied Probability*, 1(3):283–306, 1999.

R. Sutton and A. Barto. *Reinforcement learning: An introduction*. MIT Press, 1998.

Y. W. Teh, A. H. Thiery, and S. J. Vollmer. Consistency and fluctuations for stochastic gradient Langevin dynamics. *The Journal of Machine Learning Research*, 17(1):193–225, 2016.

A. W. Van der Vaart. *Asymptotic Statistics*, volume 3. Cambridge University Press, 2000.

H. Varian. Revealed preference and its applications. *The Economic Journal*, 122(560):332–338, 2012.

M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 681–688, 2011.

G. Yin and K. Yin. Passive stochastic approximation with constant step size and window width. *IEEE transactions on automatic control*, 41(1):90–106, 1996.

G. Yin, V. Krishnamurthy, and C. Ion. Regime switching stochastic approximation algorithms with application to adaptive discrete stochastic optimization. *SIAM Journal on Optimization*, 14(4): 117–1215, 2004.

G. Yin, C. Ion, and V. Krishnamurthy. How does a stochastic optimization/approximation algorithm adapt to a randomly evolving optimum/root with jump Markov sample paths. *Mathematical programming B. (Special Issue dedicated to B.T. Polyak's 70th Birthday)*, 120(1):67–99, 2009.

G. Yin, Q. Yuan, and L. Y. Wang. Asynchronous stochastic approximation algorithms for networked systems: regime-switching topologies and multiscale structure. *Multiscale Modeling & Simulation*, 11(3):813–839, 2013.

G. Yin and Q. Zhang. *Continuous-time Markov chains and applications: a two-time-scale approach*, volume 37. Springer Science & Business Media, 2013.

B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.

Appendix A. Matlab Source Code for IRL Algorithm (2) in inverse Bayesian learning of Sec.3.1. (Generates Figure 3)

```
% IRL algorithm for mutimodal mixture
mixture_weight = 0.5; nsamples=100; sigp1=sqrt(10); sigp2 = 1;  sigl1 = sqrt(2);
      sigl2 = sqrt(2); mixture_mean1 = 0; mixture_mean2 = 1;
T = 80000000; th=[0;0];  alfa = randn(2,1);
step =1e-3;  lang_step1 = 1e-5; lang_step2 =  sqrt(lang_step1);  kernelstep=0.2;
Cker = 1/(2*pi*kernelstep^2); kernelstepsq = 2*kernelstep^2;
alphaIRL = zeros(2,T); est=zeros(2,T);

for iter = 1: T
   th = randn(2,1);

  % simulate data
       if rand <  mixture_weight
           y = sigl1 * randn + mixture_mean1;
     else
          y = sigl2 * randn + mixture_mean1 + mixture_mean2;
     end;

   t1 = th(1); t2 = th(2);

% evaluate gradients
    grad1 = nsamples*((mixture_weight*exp(-(t1 - y)^2/(2*sigl1^2))*(2*t1 - 2*y))
        /(2*sigl1^3) - (exp(-(t1 + t2 - y)^2/(2*sigl2^2))*(mixture_weight - 1)
        *(2*t1 + 2*t2 - 2*y))/(2*sigl2^3))/((exp(-(t1 + t2 - y)^2/(2*sigl2^2))*(
        mixture_weight - 1))/sigl2 - (mixture_weight*exp(-(t1 - y)^2/(2*sigl1^2)
        ))/sigl1) - t1/(sigp1^2);

 grad2 = - t2/(sigp2^2) - nsamples*(exp(-(t1 + t2 - y)^2/(2*sigl2^2))*(
     mixture_weight - 1)*(2*t1 + 2*t2 - 2*y))/(2*sigl2^3*((exp(-(t1 + t2 - y)
     ^2/(2*sigl2^2))*(mixture_weight - 1))/sigl2 - (mixture_weight*exp(-(t1 - y)
     ^2/(2*sigl1^2)))/sigl1)) ;


% passive Langevin dynamics
gaus =  exp(-alfa(1)^2/2) *  exp(-alfa(2)^2/2) / ( 2 * pi);
alfa = alfa + Cker*exp( -(norm(th-alfa))^2/kernelstepsq)* (lang_step1/2  * [
    grad1;grad2]/gaus) +  lang_step2  * randn(2,1);
alphaIRL(:,iter) = alfa;

% classical stochastic gradient
th = th + step * [grad1; grad2];
est(:,iter) = th;
end;
% Plotting
figure(4);  histogram2(alphaIRL(1,5000:end),alphaIRL(2,5000:end), 'Normalization
    ','probability');
axis([-3,3,-3,3]);  xlabel('$\theta(1)$','Interpreter','latex','FontSize',18);
      ylabel('$\theta(2)$','Interpreter','latex','FontSize',18);
 figure(5);  [M1,N1] = hist3(alphaIRL(:,5000:end)',[20,20]);
contour(N1{1}, N1{2}, M1'); axis([-3,3,-3,3]);
 xlabel('$\theta(1)$','Interpreter','latex','FontSize',18);
```

```
41    ylabel('$\theta(2)$','Interpreter','latex','FontSize',18); grid on; colormap(
          jet);
```

Appendix B. Matlab Source Code for multi-kernel IRL (20) for Logistic Regression in Sec. 3.2

```
1  %  Multi-kernel IRL  for logistic  regression a9a dataset
2
3  %Algorithm parameters
4   lang_step1 = 0.25e-3; lang_step2 = sqrt(lang_step1);  sigma_theta=1;
5   sigma_kernel = 0.1;  L =100; incrementsig = 0.1;
6
7  % Read a9a Dataset
8   load datasetFeatures.mat;  load datasetLabels.mat;
9
10
11  thdim=size(features,1) + 1;  T = 32400; ; Nsweep =10; nsamples=10;
12  labels(labels<0) = 0;    % set all -1 to 0
13
14 est  = zeros(thdim,T*Nsweep); th = zeros(thdim,1); alfa = th;
15
16 for sweep = 1: Nsweep
17
18     for iter=1:T
19
20            th =   sigma_theta * randn(thdim,L); % RL chooses th randomly
21             d = vecnorm(th-alfa);
22             weight =10^(2*thdim)*exp(- d.^2/(2*sigma_kernel));
23
24             if   (sum(weight) < 1e-60)
25                    alfa = 0.1*randn(thdim,1);  %reset alfa if stuck
26             end;
27             nweight = weight/sum(weight);
28
29
30              psi =  [1;features(:,iter)]; y = labels(iter);
31              sigmoidy = 1./(1 + exp(-psi'*th));
32              wgrad =  (nsamples*psi .* (y - sigmoidy) - sign(th))*nweight';
33
34
35              k = (sweep-1) * T +iter;  k,
36
37 % logistic regression step for IRL algorithm
38              alfa = alfa +0.5*lang_step1 * wgrad +  lang_step2  * randn(thdim,1);
39
40              est(:,k) = alfa ;
41        end;
42 end;
43
44 figure(4); histogram(est(1,:),'Normalization','probability');
45 title('Multi-kernel algorithm')
```

Remarks: Out of 10 sweeps, where each sweep has 32000 iterations, only 14 resets (line 25) were required for $L = 100$ in IRL algorithm (20).

Appendix C. Matlab Source Code for multi-kernel IRL (54) to solve Constrained MDP in Sec.3.3. (Generates Fig.6(b) and (c))

```
% Multi-kernel IRL for MDP

T = 150000;  gridpoints = 100;
tp(:,:,1) = [0.8 0.2; 0.3 0.7];  tp(:,:,2) = [0.6 0.4; 0.1 0.9];
statedim=2; actiondim = 2;
 lang_step1 =5e-6;     lang_step2 =sqrt(lang_step1);
  l_step = 1/2*lang_step1;

cost = [1 100; 30 2];  constraint  = [0.2 0.3; 2 1];   lambda=1e5;
inversestep = gridpoints/2;

  pol=zeros(statedim,gridpoints^2);  Penalty_Reward = zeros(gridpoints^2,1);
   pol_con=zeros(gridpoints^2,1);
  pol_eval= zeros(gridpoints,gridpoints); alfa = zeros(2,T); cond_prob =zeros(2,
     T);

% Solve avg cost MDP exactly, over a grid of 100x100 possible randomized
    policies
for i=1:gridpoints-1,
  for j=1:gridpoints-1,

    k =gridpoints*(i-1)+j;

    policy = [i/gridpoints , 1 - i/gridpoints; j/gridpoints, 1-j/gridpoints];
    pol(:,k) = [policy(1,1); policy(2,1)];

   [ polval,polcon]  = mdp_barrier(cost,constraint,policy,tp);  % external
       function
   pol_con(k) = polcon;
    Penalty_Reward(k) = polval  - lambda * (( 1 - polcon)^2);
    pol_eval(i,j) = Penalty_Reward(k);
 end;
end

figure(7); stem3(pol(1,:),pol(2,:),Penalty_Reward,'MarkerFaceColor','g')
xlabel('pol1'); ylabel('pol2');

% Evaluate and store finite difference gradients of MDP over a 100x100 grid
for i=2:gridpoints-1,
  for j=2:gridpoints-1,
       grad(i,j,1) = (pol_eval(i+1,j) - pol_eval(i-1,j)) * inversestep;
       grad(i,j,2) = (pol_eval(i,j+1) - pol_eval(i,j-1)) * inversestep;
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% IRL algorithm
kernelstep=0.1;
 alfabar = [1;1]; Cker = 1/(2*pi*kernelstep^2);   L=50;

 for k=1:T,
    thbar =  pi/2* rand(2,L);
    d = vecnorm(alfabar-thbar);
    weight =10*exp(- d.^2/(2*kernelstep^2));
       if   (sum(weight) < 1e-6)
                         alfabar = 0.1*randn(2,1);  %reset alfa if stuck
             end;
             nweight = weight/sum(weight);
        p = (sin(thbar)).^2;
    pindex = min(max(round(gridpoints* p), [1;1]),[gridpoints -1;gridpoints -1]);
     ghatp = zeros(2,1);
    for  i=1:L
       ghatp  = [grad(pindex(1,i),pindex(2,i),1); grad(pindex(1,i),pindex(2,i)
           ,2)] .* sin(thbar(:,i)).*cos(thbar(:,i)) *  nweight(i) + ghatp ;
    end;
   ghatm = 2* ghatp;  % weighted gradient
  % Passive Langevin dynamics
   alfabar = alfabar +  (l_step * ghatm) +  lang_step2  * randn(2,1);  alfabar =
        abs(alfabar);
   alfa(:,k) = alfabar;
   cond_prob(:,k) = (sin(alfabar)).^2;   %policy conditional probabilities
  end;
% plot log of empirical density
[M,N] =  hist3(cond_prob(:,T/2:end)',[100,100]);
figure(3); stem3(flip(N{1}),flip(N{2}),(log(M)),'MarkerFaceColor','g');
xlabel('pol1');ylabel('pol2');
```

External Function used in above program

```
function [avg_cost,avg_constraint] = mdp_barrier(cost,constraint,pol,tp)
% evaluate MDP policy for average cost MDP

statedim=size(cost,1);   actiondim=size(cost,2);

%tp(:,:,1) = [0.8 0.2; 0.3 0.7];  tp(:,:,2) = [0.6 0.4; 0.1 0.9];
%pol = [0.8, 0.2; 0.3 0.7];

%cost = [1 10; 3 2];

for i=1:statedim
   for a=1:actiondim
      l= a + (i-1)*actiondim;
         for j = 1:statedim
            for abar = 1:actiondim
               m = abar + (j-1)*actiondim;
                tp_composite(l,m) = tp(i,j,a) * pol(j,abar);
             end;
         end;
         cost_vector(l) = cost(i,a);
         constraint_vector(l) = constraint(i,a);
    end;
end;

[aa,bb] =  eig(tp_composite');
joint_prob = aa(:,1)/sum(aa(:,1));

avg_constraint = constraint_vector * joint_prob;

 avg_cost = cost_vector * joint_prob;
```