Proceedings of Machine Learning Research vol 120:1–11, 2020

Policy Learning of MDPs with Mixed Continuous/Discrete Variables: A Case Study on Model-Free Control of Markovian Jump Systems

Joao P. Jansch-Porto JANSCHP2@ILLINOIS.EDU
Bin Hu BINHU7@ILLINOIS.EDU
Geir E. Dullerud DULLERUD@ILLINOIS.EDU
Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana, Illinois 61801 USA

Abstract

Markovian jump linear systems (MJLS) are an important class of dynamical systems that arise in many control applications. In this paper, we introduce the problem of controlling unknown (discrete-time) MJLS as a new benchmark for policy-based reinforcement learning of Markov decision processes (MDPs) with mixed continuous/discrete state variables. Compared with the traditional linear quadratic regulator (LQR), our proposed problem leads to a special hybrid MDP (with mixed continuous and discrete variables) and poses significant new challenges due to the appearance of an underlying Markov jump parameter governing the mode of the system dynamics. Specifically, the state of a MJLS does not form a Markov chain and hence one cannot study the MJLS control problem as a MDP with solely continuous state variable. However, one can augment the state and the jump parameter to obtain a MDP with a mixed continuous/discrete state space. We discuss how control theory sheds light on the policy parameterization of such hybrid MDPs. Then we modify the widely used natural policy gradient method to directly learn the optimal state feedback control policy for MJLS without identifying either the system dynamics or the transition probability of the switching parameter. We implement the (data-driven) natural policy gradient method on different MJLS examples. Our simulation results suggest that the natural gradient method can efficiently learn the optimal controller for MJLS with unknown dynamics.

1. Introduction

Reinforcement learning (RL) (Sutton and Barto, 2018) provides a powerful framework for solving Markov decision process (MDP) problems. Although deep RL has achieved promising empirical successes in a variety of applications (Schulman et al., 2015b; Levine et al., 2016), how to choose RL algorithms (Duan et al., 2016; Kakade, 2002; Schulman et al., 2015a; Peters and Schaal, 2008; Schulman et al., 2017) for a specific task is still not fully understood (Henderson et al., 2018; Rajeswaran et al., 2017). This motivates many recent research efforts on understanding the performances of RL algorithms on simplified benchmarks. For example, many control applications lead to MDP formulations with continuous state/action spaces, and hence the classic Linear Quadratic Regulator (LQR) problem has been revisited as a benchmark for understanding the performance of various model-free or model-based RL algorithms on such MDPs (Fazel et al., 2018; Dean et al., 2017; Malik et al., 2018; Tu and Recht, 2018; Dean et al., 2018; Abbasi-Yadkori and Szepesvári, 2011; Abbasi-Yadkori et al., 2018; Yang et al., 2019).

Another important class of MDPs involve mixed continuous/discrete state variables (Boyan and Littman, 2001; Toussaint and Storkey, 2006; Guestrin et al., 2004). In this paper, our key point is that the problem of controlling unknown (discrete-time) Markov Jump Linear Systems (MJLS) (Costa et al., 2006) provides a simple meaningful benchmark for understanding the performances of policy-based RL algorithms on MDPs with mixed continuous/discrete variables. MJLS is an important

class of dynamical systems that find many applications in control (Bar-Shalom and Li, 1993; Fox et al., 2011; Gopalakrishnan et al., 2016; Pavlovic et al., 2000; Sworder and Boyd, 1999; Vargas et al., 2013), and machine learning (Hu et al., 2017; Hu and Syed, 2019). Notice that the state/input matrices of a MJLS are functions of a jump parameter that is typically sampled from a Markov chain with discrete state variables. Understanding the performance of various RL methods on the MJLS control problem can bring many useful insights. An important fact is that the state of a MJLS does not form a Markov chain and hence one cannot study the MJLS control problem as a MDP with solely continuous state variable. However, if one augments the state and the jump parameter together, a MDP with a mixed continuous/discrete state space is naturally obtained. Therefore, we believe the optimal control of unknown MJLS is a meaningful benchmark for further understanding of mixed MDPs.

Although the MDP formulation for MJLSs involves mixed continuous/discrete state variables, our study demonstrates that we can still modify policy-based RL algorithms, such as REINFORCE (Williams, 1992; Sutton et al., 2000), to efficiently solve this problem. The key here is to parameterize the control policy based on MJLS control theory. Recently, model-based policy optimization methods have been shown to provably converge to the global optimal policy for MJLS control (Jansch-Porto et al., 2020). In this paper, we discuss how to efficiently implement these methods in a data-driven manner. We implement the model-free natural policy gradient (NPG) method on various MJLS examples. Our simulation results suggest that the NPG method with the REINFORCE policy gradient estimator and a simple average baseline can learn the optimal control for an MJLS without identifying either the system dynamics or the transition probability of the jump parameter. This confirms that policy optimization may provide a promising solution for controlling unknown jump systems. It is our hope that our study serves a first step towards more understanding of RL algorithms on MDPs with mixed continuous/discrete variables.

Related work. Some previous work has studied how to apply RL methods to MJLSs with known state/input matrices and unknown jump parameter distribution (Costa and Aya, 2002; Beirigo et al., 2018). In our paper, both state/input matrices and the transition probability of the jump parameter are assumed to be unknown. In addition, the continuous-time setup has also been investigated recently (He et al., 2019). Our paper considers the standard discrete-time formulation of MJLSs.

2. Background and Preliminaries

2.1. Notation

We denote the set of real numbers by $\mathbb{R}$. Let Z be a square matrix, and we use the notation Z^T, $\|Z\|$, $\mathrm{tr}(Z)$, $\sigma_{\min}(Z)$ to denote its transpose, spectral norm, trace, and minimum singular value, respectively. We indicate positive definite matrices by $Z \succ 0$. Given matrices $\{D_i\}_{i=1}^m$, let $\mathrm{diag}(D_1, \ldots, D_m)$ denote the block diagonal matrix whose (i,i)-th block is D_i. An identity matrix of dimension n is denoted by I_n. We use $\otimes$ to denote the Kronecker product, and $\mathrm{vec}(X)$ to denote the vectorization of the matrix X formed by stacking the columns of X into a single column. We use $\hat{e}_i$ to denote the canonical basis vector in $\mathbb{R}^n$, where the only nonzero entry is the index i. The normal distribution with mean $m \in \mathbb{R}^n$ and covariance $\Lambda \in \mathbb{R}^{n \times n}$ is denoted by $\mathcal{N}(m, \Lambda)$.

2.2. Markovian Jump Linear Systems

A Markovian jump linear system is governed by the discrete-time state-space model

$$x_{t+1} = A_{\omega_t} x_t + B_{\omega_t} u_t + e_t, \text{ with } x_0 \sim \mathcal{D} \text{ and } e_t \sim \mathcal{N}(0, \varepsilon^2 I), \tag{1}$$

where $x_t \in \mathbb{R}^d$ and $u_t \in \mathbb{R}^k$ correspond to the state and control action at time $t \in \mathbb{N}_0$, respectively. The matrices $A_{\omega_t} \in \mathbb{R}^{d\times d}$ and $B_{\omega_t} \in \mathbb{R}^{d\times k}$ depend on a jump parameter ω_t, which is sampled from a Markov chain with a discrete state space $\Omega := \{1, \ldots, n_s\}$. Hence we have $A_{\omega_t} \in \{A_i\}_{i\in\Omega}$ and $B_{\omega_t} \in \{B_i\}_{i\in\Omega}$. Denote the transition probabilities and initial distribution of ω_t as $p_{ij} = \mathbb{P}(\omega_{t+1} = j | \omega_t = i)$ and $\rho = \begin{bmatrix}\rho_1 & \cdots & \rho_{n_s}\end{bmatrix}^T$. We have $\sum_{j=1}^{n_s} p_{ij} = 1$, and $\sum_{i\in\Omega} \rho_i = 1$.

In this paper, we are interested in minimizing the following discounted quadratic cost

$$C = \mathbb{E}_{x_0\sim\mathcal{D},\omega_0\sim\rho}\left[\sum_{t=0}^{\infty} \gamma^t (x_t^T Q_{\omega_t} x_t + u_t^T R_{\omega_t} u_t)\right], \tag{2}$$

where $Q_{\omega_t} \succ 0$, $R_{\omega_t} \succ 0$ and $\gamma \in (0, 1)$. When the model information is available, the above MJLS LQR problem can be solved using standard Algebraic Riccati Equation (ARE) techniques (Fragoso, 1989). Specifically, let $\{P_i\}_{i\in\Omega}$ be the positive definite solution to the following coupled AREs:

$$P_i = Q_i + \gamma A_i^T \mathcal{E}_i(P) A_i - \gamma^2 A_i^T \mathcal{E}_i(P) B_i \left(R_i + \gamma B_i^T \mathcal{E}_i(P) B_i\right)^{-1} B_i^T \mathcal{E}_i(P) A_i, \tag{3}$$

where $\mathcal{E}_i(P) := \mathbb{E}\left[P_{\omega_{t+1}} \middle| \omega_t = i\right] = \sum_{j=1}^{n_s} p_{ij} P_j$. It is known that the optimal cost can be achieved using a state-feedback controller $u_t = -K^*_{\omega_t} x_t$ where $K_i^* = \gamma \left(R_i + \gamma B_i^T \mathcal{E}_i(P) B_i\right)^{-1} B_i^T \mathcal{E}_i(P) A_i$. In this paper, we are interested in model-free learning of K_i^* for the case where the model parameters A_i, B_i, Q_i, R_i, and p_{ij} are unknown.

2.3. A Brief Review of Policy Learning for LTI Systems

Here we briefly review model-free policy learning for LTI systems. LTI systems are a special case of MJLS, where $\Omega = \{1\}$. When applying policy-based RL methods for LTI systems, one first needs to specify the policy parameterization. Since the state and action spaces are continuous, it is quite natural to adopt the linear Gaussian policy $u_t \sim \mathcal{N}(-Kx_t, \sigma^2 I)$, where K and σ are the parameters to be learned from data. Then, it is straightforward to apply REINFORCE or natural policy gradient to update (K, σ). In general, it is difficult to obtain finite sample guarantees for REINFORCE and its variants. In Fazel et al. (2018), it is shown that the population dynamics of the NPG method has linear convergence to the optimal policy if a stabilizing initial policy is used. In addition, the authors also present a finite sample analysis of the model-free zeroth-order optimization (Conn et al., 2009; Nesterov and Spokoiny, 2017) implementation of the NPG method. Notice that zeroth-order optimization (or evolutionary strategies) does not require a stochastic policy for exploration and hence the authors consider a deterministic policy. Consequently, their finite sample analysis cannot be directly extended for REINFORCE. Nevertheless, it is expected that REINFORCE will work for the LTI problem as long as the gradient estimations are reasonably close to the true gradient.

3. Policy Learning for MJLS

Now we recast the MJLS LQR problem within the RL framework. This formulation involves a MDP with mixed continuous/discrete state variables. For the MJLS (1), the system state $\{x_t\}$ itself

does not form a Markov chain anymore. However, if we augment (x_t, ω_t) as the new state, we will obtain a hybrid MDP with mixed continuous/discrete state variables. Now, the full state space is the product $\mathbb{R}^{n_x} \times \Omega$, but action space is still $\mathbb{R}^{n_u}$. The joint state transition model is specified by both the transition probability $\{p_{ij}\}$ and the MJLS model (1).

The above MDP adopts a model-based dynamic programming solution which is summarized as the ARE approach reviewed in Section 2.2. The model-based approach requires knowing the model parameters $(A_i, B_i, Q_i, R_i, p_{ij})$ in advance. Alternatively, when the model is unknown, one can apply model-free RL algorithms to solve this problem. This makes the MJLS LQR problem arguably the most basic benchmark for RL with hybrid MDPs. Understanding the performance of RL algorithms on this benchmark may shed light on how to solve more complicated hybrid MDPs. In this paper, we focus on applying policy-based RL methods for solving such hybrid MDPs.

3.1. Policy Parameterization and Optimization Landscape

To apply policy optimization for the above MJLS MDP, we need to confine the search to some certain class of policies. Recently the important role of the policy representation has been recognized. For hybrid MDPs. there are multiple choices for policy parameterization. One choice is to adopt a neural network structure where both x_t and ω_t are fed as inputs. However, the optimization landscape for such a neural network parameterization is unclear. Another choice is based on optimal control theory for MJLS. Since we know the the optimal cost for the MJLS LQR problem can be achieved by a control law in the form of $u_t = -K_{\omega_t} x_t$, it is reasonable to restrict the policy search within the class of state feedback controllers in the form of $u_t \sim \mathcal{N}(-K_{\omega_t} x_t, \sigma^2 I)$. Specifically, we can set $\hat{K} = \begin{bmatrix} K_1 & \cdots & K_{n_s} \end{bmatrix}$, where K_i is the feedback gain for mode i. With this notation, we obtain a policy optimization problem whose decision variables are $\hat{K}$ and σ.

Our policy parameterization can be thought as a mixture of continuous Gaussian policy and discrete look-up tables. For each of system modes, we train a corresponding linear policy specified by K_i. Eventually there are n_s different linear policies stored in a look-up table for various possible values of ω_t. One advantage of our parameterization is that it becomes clear that the cost function $C(\hat{K}, \sigma)$ only has one stationary point which is the global minimum for the MJLS LQR problem. To see this, we first write down an analytical formula for $\nabla C(\hat{K}, \sigma)$. Let $P_i^{\hat{K}}$ denote the solution to the coupled Lyapunov equations:

$$P_i^{\hat{K}} = Q_i + K_i^T R_i K_i + \gamma \left(A_i - B_i K_i\right)^T \mathcal{E}_i(P^{\hat{K}}) \left(A_i - B_i K_i\right), \text{ for } i \in \Omega. \tag{4}$$

Then the cost (2) subject to the system dynamics (1) and the Gaussian policy $u_t \sim \mathcal{N}(-K_{\omega_t} x_t, \sigma^2 I)$ can be calculated as

$$C(\hat{K}, \sigma) = \mathbb{E}_{x_0 \sim \mathcal{D}} \left[\sum_{i \in \Omega} \rho_i \left(x_0^T P_i^{\hat{K}} x_0 + z_i \right) \right], \tag{5}$$

with $z_i = \sigma^2 \text{tr}\big(R_i + \gamma B_i^T \mathcal{E}_i(P) B_i\big) + \varepsilon^2 \text{tr}(\mathcal{E}_i(P)) + \gamma \sum_{j \in \Omega} p_{ij} z_j$. Denote $X_i(t) := \mathbb{E}\left[x_t x_t^T \mathbf{1}_{\omega_t = i}\right]$. Then we can calculate the policy gradient $\nabla C(\hat{K}, \sigma)$ using the following explicit formula.

Lemma 1 *Given $\hat{K}$ stabilizing (1) in the mean square sense and $\sigma \geq 0$, the gradient of (5) with respect to control gain parameters $\hat{K}$ and noise level σ is given as*

$$\nabla C(\hat{K}, \sigma) = \begin{bmatrix} \text{vec}\big(2 \begin{bmatrix} L_1(\hat{K}) & L_2(\hat{K}) & \cdots & L_{n_s}(\hat{K}) \end{bmatrix} \chi_{\hat{K}}\big) \\ \frac{\sigma}{1-\gamma} \sum_{j=1}^{n_s} \rho_i \ \text{tr}\big(R_i + \gamma B_i^T \mathcal{E}_i(P^{\hat{K}}) B_i\big) \end{bmatrix} = \begin{bmatrix} \text{vec}(F_K) \\ F_\sigma \end{bmatrix} \tag{6}$$

where $L_i(\hat{K}) = \left(R_i + \gamma B_i^T \mathcal{E}_i(P^{\hat{K}})B_i\right)K_i - \gamma B_i^T \mathcal{E}_i(P^{\hat{K}})A_i$, *and*

$$\chi_{\hat{K}} = \sum_{t=0}^{\infty} \gamma^t \text{diag}\left(X_1(t), \ldots, X_{n_s}(t)\right). \tag{7}$$

Later, for simplicity, we use $\nabla_{\hat{K}} C(\hat{K}, \sigma) \coloneqq \nabla C(\hat{K}) = F_K$ *and* $\nabla_{\hat{\sigma}} C(\hat{K}, \sigma) \coloneqq \nabla C(\sigma) = F_\sigma$[1].

Proof The differentiability of $C(\hat{K}, \sigma)$ can be proved using the implicit function theorem, and this step is similar to the proof of Lemma 3.1 in Rautert and Sachs (1997). The derivation of the gradient formula follows the similar steps to Lemma 1 in Jansch-Porto et al. (2020). ■

Suppose $\mathbb{E}_{x_0 \sim \mathcal{D}}\left[x_0 x_0^T\right]$ is full rank and $\rho_i > 0$ for all i. Then a stationary point given by $\nabla C(\hat{K}, \sigma) = 0$ has to satisfy $L_i(\hat{K}) = \left(R_i + \gamma B_i^T \mathcal{E}_i(P^{\hat{K}})B_i\right)K_i - \gamma B_i^T \mathcal{E}_i(P^{\hat{K}})A_i = 0$ for all $i \in \Omega$, and $\sigma = 0$. It becomes obvious that $L_i(\hat{K}) = 0$ leads to the global optimal policy $\hat{K}^*$ defined by (3). Hence the only stationary point is the global minimum of the original MJLS LQR problem.

Notice that the optimization of $C(\hat{K}, \sigma)$ is a constrained optimization problem whose feasible set consists of all $\hat{K}$ stabilizing the closed-loop dynamics in the mean square sense. The cost function $C(\hat{K}, \sigma)$ is finite and differentiable only within the feasible set.

3.2. Linear Convergence of the Population Dynamics of NPG

If the model information is known, one can solve $\hat{K}^*$ using the following model-based NPG updates:

$$\hat{K}^{n+1} = \hat{K}^n - \eta_n \sigma_n^2 \nabla C(\hat{K}^n) \chi_{\hat{K}^n}^{-1}, \text{ and } \sigma_{n+1} = \sigma_n - \alpha_n \sigma_n^2 \nabla C(\sigma_n) \left(\frac{1-\gamma}{2k}\right) \tag{8}$$

The initial policy is denoted as $\hat{K}^0$ which is assumed to stabilize the closed-loop dynamics in the mean square sense. In Jansch-Porto et al. (2020), it is shown that the model-based NPG updates for the MJLS LQR problem with a deterministic policy parameterization and $\gamma = 1$[2] can be guaranteed to stay in the feasible set and converge to the global minimum. We can obtain similar results for the discounted case with a Gaussian policy, and prove the global convergence of (8). We omit these proofs here. In following sections, we will consider the case where the model information is unknown, and implement the model-free natural policy gradient method whose population dynamics exactly matches the above model-based updates. We will use model-free RL techniques to estimate $\nabla C(\hat{K})$ and $\chi_{\hat{K}}$ from data. It is expected that the model-free NPG updates will closely track the dynamics of (8) and work well if sufficient data is provided for the gradient estimation. Connections between NPG and the exact dynamics (8) are further discussed in the `arXiv` version of our paper.

3.3. Natural Policy Gradient and REINFORCE

Now we discuss the model-free policy learning of the MJLS LQR problem. From the exact natural policy gradient update rule (8), we need to obtain estimates for both the policy gradient and state

1. We slightly abuse our notation here. Notice F_K is a matrix and $\nabla C(\hat{K}, \sigma)$ is a vector. When calculating F_K, we take the gradient with respect to all entries of $\hat{K}$. Then we augment the columns of F_K with F_σ to obtain $\nabla c(\hat{K}, \sigma)$.
2. In Jansch-Porto et al. (2020), model-based policy optimization is considered. One does not need a stochastic policy for exploration. Hence it is sufficient to use a deterministic policy there, and the discounted factor is not needed for the problem formulation

covariances. Based on (7), we can directly estimate $\chi_{\hat{K}}$ by averaging $\sum_{t=0}^{T_F} \gamma^t x_t x_t^T$ (with some large T_F) over multiple sampled trajectories of the MJLS model (1).

To estimate $\nabla C(\hat{K}, \sigma)$ we will adopt the REINFORCE algorithm, which uses a Monte Carlo rollout to estimate the policy gradient. Specifically, for a stochastic policy $\pi_\theta(u_t|x_t, w_t)$, we can set $\nabla C(\theta) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \nabla_\theta \log \pi_\theta(u_t|x_t, \omega_t)\Psi_t\right]$, where Ψ_t can be calculated using one of following choices: 1) total reward of the trajectory; 2) reward following input u_t; 3) baseline version of the previous item; 4) state-action value function; 5) advantage function; 6) TD residual; 7) generalized advantage estimation. In this paper, we consider the baseline approach. Let us align the notation as

$$\theta = \begin{bmatrix} \text{vec}\left(\hat{K}\right) \\ \sigma \end{bmatrix}.$$

It is known that we can add an arbitrary baseline without affecting the expectation of the policy gradient. Here we used the cumulative average reward at timestep t as our baseline. The policy gradient above is in terms of an expectation, so we can use sampling methods to approximate it as

$$\nabla C(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \left(\sum_{t=0}^{T_F} \gamma^t \nabla_\theta \log \pi_\theta(u_{t,i}|x_{t,i}, \omega_{t,i}) \left(\sum_{t'=t}^{T_F} \gamma^{t'-t} c_{t',i} - b_{t,i} \right) \right) =: \widehat{\nabla C}(\theta)$$

for sufficiently large N and T_F. Here, N is the number of sampled trajectories and T_F is the horizon length. We calculate the baseline as $b_{t,j} = \frac{1}{j} \sum_{i=1}^{j} \left(\sum_{t'=t}^{T_F} \gamma^{t'-t} c_{t',i} \right)$. We use a cumulative formulation of the baseline so it is not necessary to store all previous trajectories. Algorithm 1 below provides a procedure to compute the estimates $\widehat{\nabla C}(\hat{K})$ and $\widehat{\chi}_{\hat{K}^n}$

Having obtained a model-free policy gradient estimation, we can directly use the gradient estimates to update the controller gains with the natural policy gradient step:

$$\hat{K}^{n+1} = \hat{K}^n - \eta_n \sigma_n^2 \widehat{\nabla C}(\hat{K}^n) \widehat{\chi}_{\hat{K}^n}^{-1}, \text{ and } \sigma_{n+1} = \sigma_n - \alpha_n \sigma_n^2 \widehat{\nabla C}(\sigma_n) \left(\frac{1-\gamma}{2k} \right) \tag{9}$$

We now discuss Algorithm 1, which is repeated at every NPG iteration step. To run Algorithm 1, we need to specify the control policy $\hat{K}$, an input variance σ^2, the single trajectory length T_F, and the total number of trajectories N. For each trajectory generated we need to measure the systems states x_t, actions u_t, state-action cost c_t, and the jump parameter ω_t. We note that it is common to use a constant exploration noise σ in many RL algorithms, however here we are treating σ as a parameter one wants to learn. While we know that the optimal σ for the LQR problem is zero, having a variable σ helps finding a better trade-off between exploration/exploitation. For our simulations, we set up α_n in a way that σ decreases to 0 at a pre-specified linear rate 0.99.

4. Model-Free Implementations

In this section, we implement the model-free policy gradient algorithm to different example systems.

4.1. Small scale problem

Consider a MJLS which can switch between two modes, where each mode (A_1, B_1) and (A_2, B_2) are not individually stabilizable, but the switched system is. We define the state space matrices as:

$$A_1 = \begin{bmatrix} 0.4 & 0.6 & -0.1 \\ -0.4 & -0.6 & 0.3 \\ 0 & 0 & 1 \end{bmatrix}, \quad B_1 = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0.9 & 0.5 & -0.1 \\ 0 & 1 & 0 \\ -0.1 & 0.5 & -0.4 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix},$$

Algorithm 1: Model-Free Switched Policy Gradient Estimation

Starting from a control policy $\hat{K}$, input variance σ^2, trajectory length T_F, and batch size N;
for $i = 1, \dots, N$ **do**
  Generate a trajectory τ_i and measure $\{x_t, u_t, \omega_t, c_t\}_{t=0,\dots,T_F}$ starting from $x_0 \sim \mathcal{D}$ with $u_t \sim \mathcal{N}(-K_{\omega_t} x_t, \sigma^2 I)$;
  for $t = 0, \dots, T_F$ **do**
    Compute $\hat{G}_t = -\frac{1}{\sigma^2}(u_t + K_{\omega_t} x_t)\,(\hat{e}_{\omega_t} \otimes x_t)^T$, $\hat{Q}_t = \sum_{t'=t}^{T_F} \gamma^{t'-t} c_t$, $\hat{S}_t = -\frac{k}{\sigma} + \frac{1}{\sigma^3}(u_t + K_{\omega_t} x_t)^T (u_t + K_{\omega_t} x_t)$, $\hat{X}_{\omega_t,i} \mathrel{+}= \gamma^t x_t x_t^T$, and $b_t = ((i-1) b_t + \hat{Q}_t)/i$;
  end
  Compute $\widehat{\nabla C}_i(\hat{K}) = \sum_{t=0}^{T_F} \gamma^t \hat{G}_t (\hat{Q}_t - b_t)$ and $\widehat{\nabla C}_i(\sigma) = \sum_{t=0}^{T_F} \gamma^t \hat{S}_t (\hat{Q}_t - b_t)$;
end
Return the estimates: $\widehat{\nabla C}(\cdot) = \frac{1}{N} \sum_{i=0}^{N} \widehat{\nabla C}_i(\cdot)$ and $\widehat{\chi}_{\hat{K}^n} = \frac{1}{N} \sum_{i=0}^{N} \operatorname{diag}\big(\hat{X}_{1,i}, \dots, \hat{X}_{n_s,i}\big)$

with cost matrices $Q_1 = I_3$, $R_1 = 1$, $Q_2 = 2I_3$, and $R_2 = 2$. We also set the transition probability $\mathcal{P} = \begin{bmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{bmatrix}$, and initial distribution $\rho = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$.

Using Algorithm 1 with (9), we computed the policy of the system above. For all simulations, we set $K_1 = K_2 = 0_{k \times d}$ as our initial gain values since $C(\hat{K}^0, \sigma_0)$ is finite, and used the parameters $T = 500$, $\sigma_0 = 0.5$, $\gamma = 0.99$, and $\eta = 0.01\sigma_0^{-2}$. For $N \in \{1000, 2500, 5000, 10000\}$, we ran 100 steps of (9) and computed the associated costs of the policy obtained. Each policy iteration was computed 1000 times, with the results shown in Figure 1(*a*).

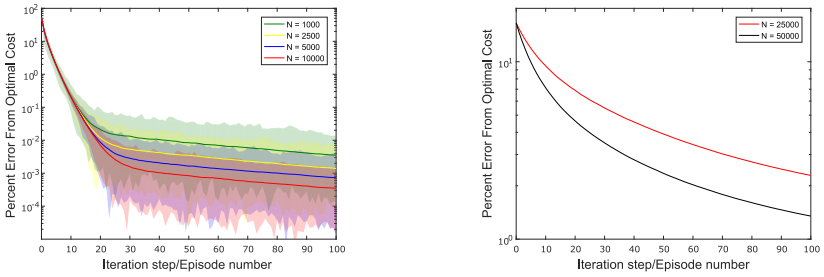
4.2. System with large number of modes

We now consider a system with 100 states, 20 inputs, and 100 modes. The matrices A and B were generated using `drss` in MATLAB in order to guarantee that the system would have finite cost with $\hat{K}^0 = 0$. The probability transition matrix $\mathcal{P}$ was sampled from a Dirichlet Process $\text{Dir}(99 \cdot I_{100} + 1)$, which always results in an irreducible Markov chain. For simplicity, we set $\rho_i = 1/100$, $Q_i = I$, and $R_i = I$ for all $i \in \Omega$. Here, we used the step size $\eta = 0.000125\sigma_0^{-2}$, initial noise $\sigma_0 = 1$, and batch sizes $N = \{25000, 50000\}$. The resulting policy costs are shown in Figure 1(*b*).

Obtaining controllers for systems with a large number of modes can be computationally hard using (3), as the number of coupled equations grows with the number of modes. By using the data-driven approach, we only need to assert that we visit each mode often enough. Here, this condition is directly satisfied since the modes are sampled from an irreducible Markov chain.

4.3. Structured Controller

Now we consider the case where we want to impose some certain structure to the designed controller. For example, in the output feedback problem, the controller cannot access the full state measurements. Structured control design also finds many applications in decentralized control, where individual controllers might not have access to the global system state. To find the optimal structured controller, we can simply project the estimated gradient to maintain the desired structure, and update the control gains using gradient descent, instead of the natural policy gradient.



(*a*) Small scale example (*b*) Large number of modes

Figure 1: Shown are the relative error between policy $\hat{K}^i$, obtained using Algorithm 1, and the optimal policy $\hat{K}^*$, computed using (3). The relative error was calculated as $\left|\frac{C(\hat{K}^i,0)-C(\hat{K}^*,0)}{C(\hat{K}^*,0)}\right| \times 100$. The solid lines indicate the mean expected percent error.

To illustrate how the projected gradient descent works for the structured control design problem, we test the algorithm on a small example system. Consider the following two-mode system:

$$A_1 = \begin{bmatrix} -0.4 & 1.0 \\ 0.0 & 0.9 \end{bmatrix}, \quad A_2 = \begin{bmatrix} 0.0 & 1.0 \\ -0.4 & 0.9 \end{bmatrix}, \quad \{B_i\}_{i=1,2} = \begin{bmatrix} 1.0 & 0.5 \\ 0.0 & 2.0 \end{bmatrix},$$

with the weighting matrices and transition probability:

$$\{Q_i\}_{i=1,2} = \begin{bmatrix} 10 & 0 \\ 0 & 20 \end{bmatrix}, \quad \{R_i\}_{i=1,2} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathcal{P} = \begin{bmatrix} 0.8 & 0.2 \\ 0.3 & 0.7 \end{bmatrix}, \quad \rho = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}.$$

Using (3) and (4), the total expected cost following the optimal unstructured policy, $\hat{K}^*_{unstruc}$, is $C(\hat{K}^*_{unstruc}) = 2.5704$, while the expected cost of having no feedback, $\hat{K}_0$, is $C(\hat{K}_0) = 8.4861$.

Now suppose that we can only measure the first state of the system. This is equivalent to having a controller of the form

$$K_{shape} = \begin{bmatrix} \bullet & 0 \\ \bullet & 0 \end{bmatrix}. \tag{10}$$

If we project the estimate of the gradient onto (10), and iterate using gradient descent, we obtain the expected cost $C(\hat{K}^{100}_{struct}) = 6.2226$ after 100 iteration steps. Clearly, the resultant structured control gain is not simply the projection of the optimal unstructured control gain onto the structured space. If we simply project $\hat{K}^*_{unstruc}$ onto (10), and denote that by $\hat{K}^*_{proj}$, the total expected cost becomes $C(\hat{K}^*_{proj}) = 13.3227$, which is worse than not having any feedback action at all.

5. Conclusion

In this paper we revisited the optimal control of Markovian Jump Linear Systems as a benchmark for further understanding of policy-based RL algorithms and hybrid MDPs. We discussed how to set up the policy parameterization for such hybrid MDPs, and present an efficient data-driven implementation of the natural policy gradient method for learning optimal state-feedback controllers of unknown MJLSs. We demonstrated the performance of the model-free natural policy gradient method on different example systems. Our results suggest that it is promosing to apply policy-based RL methods for optimal control of large scale switching systems, where the computational complexity grows as the system size increases.

References

Y. Abbasi-Yadkori and C. Szepesvári. Regret bounds for the adaptive control of linear quadratic systems. In *Proceedings of the 24th Annual Conference on Learning Theory*, pages 1–26, 2011.

Y. Abbasi-Yadkori, N. Lazic, and C. Szepesvári. Regret bounds for model-free linear quadratic control. *arXiv preprint arXiv:1804.06021*, 2018.

Y. Bar-Shalom and X. Li. Estimation and tracking- principles, techniques, and software. *Norwood, MA: Artech House, Inc, 1993.*, 1993.

Rafael L Beirigo, Marcos Garcia Todorov, and André da Motta Salles Barreto. Online TD (λ) for discrete-time Markov jump linear systems. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 2229–2234, 2018.

J. A Boyan and M. L. Littman. Exact solutions to time-dependent MDPs. In *Advances in Neural Information Processing Systems*, pages 1026–1032, 2001.

A. Conn, K. Scheinberg, and L. Vicente. *Introduction to derivative-free optimization*, volume 8. Siam, 2009.

O. Costa, M. Fragoso, and R. Marques. *Discrete-time Markov jump linear systems*. Springer London, 2006.

Oswaldo LV Costa and Julio CC Aya. Monte Carlo TD (λ)-methods for the optimal control of discrete-time Markovian jump linear systems. *Automatica*, 38(2):217–225, 2002.

S. Dean, H. Mania, N. Matni, B. Recht, and S. Tu. On the sample complexity of the linear quadratic regulator. *arXiv preprint arXiv:1710.01688*, 2017.

S. Dean, H. Mania, N. Matni, B. Recht, and S. Tu. Regret bounds for robust adaptive control of the linear quadratic regulator. In *Advances in Neural Information Processing Systems*, pages 4188–4197, 2018.

Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pages 1329–1338, 2016.

M. Fazel, R. Ge, S. Kakade, and M. Mesbahi. Global convergence of policy gradient methods for the linear quadratic regulator. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 1467–1476, 2018.

E. Fox, E.B. Sudderthand M.I. Jordan, and A.S. Willsky. Bayesian nonparametric inference of switching dynamic linear models. *IEEE Transactions on Signal Processing*, 59(4):1569 – 1585, 2011.

M. Fragoso. Discrete-time jump LQG problem. *International Journal of Systems Science*, 20(12): 2539–2545, 1989.

K. Gopalakrishnan, H. Balakrishnan, and R. Jordan. Stability of networked systems with switching topologies. In *IEEE Conference on Decision and Control*, pages 1889–1897, 2016.

C. Guestrin, M. Hauskrecht, and B. Kveton. Solving factored MDPs with continuous and discrete variables. In *Proceedings of the 20th conference on Uncertainty in Artificial Intelligence*, pages 235–242, 2004.

Shuping He, Maoguang Zhang, Haiyang Fang, Fei Liu, Xiaoli Luan, and Zhengtao Ding. Reinforcement learning and adaptive optimization of a class of Markov jump systems with completely unknown dynamic information. *Neural Computing and Applications*, pages 1–10, 2019.

P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger. Deep reinforcement learning that matters. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

B. Hu and U. Syed. Characterizing the exact behaviors of temporal difference learning algorithms using Markov jump linear system theory. In *Advances in Neural Information Processing Systems*, pages 8477–8488, 2019.

B. Hu, P. Seiler, and A. Rantzer. A unified analysis of stochastic optimization methods using jump system theory and quadratic constraints. In *Conference on Learning Theory*, pages 1157–1189, 2017.

J.P. Jansch-Porto, B. Hu, and G.E. Dullerud. Convergence guarantees of policy optimization methods for markovian jump linear systems. *accepted to ACC*, 2020.

S. Kakade. A natural policy gradient. In *Advances in neural information processing systems*, pages 1531–1538, 2002.

S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

D. Malik, A. Pananjady, K. Bhatia, K. Khamaru, P. Bartlett, and M. Wainwright. Derivative-free methods for policy optimization: Guarantees for linear quadratic systems. *arXiv preprint arXiv:1812.08305*, 2018.

Y. Nesterov and V. Spokoiny. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17(2):527–566, 2017.

V. Pavlovic, J.M. Rehg, and J. MacCormick. Learning switching linear models of human motion. In *Advances in Neural Information Processing Systems*, 2000.

J. Peters and S. Schaal. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 2008.

A. Rajeswaran, K. Lowrey, E. Todorov, and S. Kakade. Towards generalization and simplicity in continuous control. In *Advances in Neural Information Processing Systems*, pages 6550–6561, 2017.

T. Rautert and E. Sachs. Computational design of optimal output feedback controllers. *SIAM Journal on Optimization*, 7(3):837–852, 1997.

J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015a.

J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representation*, 2015b.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

R. Sutton and A. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

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

D. Sworder and J. Boyd. *Estimation problems in hybrid systems*. Cambridge University Press, 1999.

M. Toussaint and A. Storkey. Probabilistic inference for solving discrete and continuous state markov decision processes. In *Proceedings of the 23rd international conference on Machine learning*, pages 945–952. ACM, 2006.

S. Tu and B. Recht. The gap between model-based and model-free methods on the linear quadratic regulator: An asymptotic viewpoint. *arXiv preprint arXiv:1812.03565*, 2018.

A. N. Vargas, E. F. Costa, and J. B. R. do Val. On the control of Markov jump linear systems with no mode observation: Application to a DC motor device. *International Journal of Robust and Nonlinear Control*, 23(10):1136–1150, 2013.

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

Z. Yang, Y. Chen, M. Hong, and Z. Wang. On the global convergence of actor-critic: A case for linear quadratic regulator with ergodic cost, 2019.