Reinforcement Learning for Partially Observable Linear Gaussian Systems Using Batch Dynamics of Noisy Observations

Farnaz Adib Yaghmaie, Hamidreza Modares, and Fredrik Gustafsson, *Fellow, IEEE*

***Abstract*—Reinforcement learning algorithms are commonly used to control dynamical systems with measurable state variables. If the dynamical system is partially observable, reinforcement learning algorithms are modified to compensate for the effect of partial observability. One common approach is to feed a finite history of input–output data instead of the state variable. In this article, we study and quantify the effect of this approach in linear Gaussian systems with quadratic costs. We coin the concept of *L-Extra-Sampled-dynamics* to formalize the idea of using a finite history of input–output data instead of state and show that this approach increases the average cost.**

***Index Terms*—Linear quadratic Gaussian, partiially observable dynamical systems, reinforcement learning.**

I. Introduction

Reinforcement learning (RL) is one of the main branches of machine learning where a decision-making center, (the agent) tries to find an optimal policy through interaction with the environment. One possible way to categorize RL agents is based on the function they are approximating [1], [2], given in the following:

1) *dynamic programming-based solutions:* approximate the value functions by minimizing the Bellman error [3] in the *temporal difference* learning context;
2) *policy search solutions:* directly optimize the performance index by learning a parametric policy [4], [5], [6];
3) *model-building solutions:* estimate a model of the environment [7], [8] and then solve the optimal control problem for the estimated model.

This concept is known as adaptive control [9], and there is vast literature around it. It is worth noting that these categories are not mutually exclusive, and many modern RL algorithms combine multiple approaches to leverage their respective advantages.

In a typical RL setting, the environment is represented by a Markov decision process (MDP) with unknown dynamics. In practice, the measurement of the state variable might be noisy [10], [11] or the sensor data might not contain all information about the state variable. A valid representation for the environment, in this case, is a *partially observable MDP* or partially observable dynamical systems. There are a number of ways to tackle the problem of partial observability in RL frameworks, all centering around how to use the past data to extract information about the dynamics, state, and noise. One possibility is to use recurrent structures, such as long short term memory units, and integrate arbitrary long histories of data [12], [13], [14]. The *Lth-order history approach* [4] is an another way of handling partial observability where the idea is to feed a finite history of input–output data to the RL algorithm. To study the effect of such approaches, one needs to consider a simplified setup where analytical analysis is possible.

The *linear quadratic (LQ)* problem is a good benchmark as it is possible to solve the problem analytically. In an LQ problem, a dynamical system is linear and the state variable is only partially observable and noisy. The performance index to be minimized in the LQ problem is quadratic. In [15] and [16], input–output data history is used to estimate the Markov parameters of linear dynamics with stochastic or adversarial noises, and the controller is parameterized based on the denoised observations. For deterministic systems with unmeasured states and measured outputs, a long history of input–output data is used in [17] to learn the value function. However, learning input–output-based control policies for stochastic systems with dynamics and measurement noises is unsettled and requires new developments to account for noises.

In this article, we consider linear Gaussian dynamical systems in which the state variable is only partially observable and noisy. We assume that the dynamics of the system is unknown. Since the system is corrupted by both process and observation noise, it is imperative to account for the effect of using a finite history of corrupted input–output data on the average cost without resorting to filtering when learning an optimal controller using data. This is a common practice in complex environments where building a model of the environment and estimating the state are difficult tasks. To formalize learning an optimal controller under noisy data, we introduce the concept of *L-Extra-Sampled (Les)-dynamics* for stochastic systems. Based on Les-dynamics, we show that one can define a point estimator of the state variable using the history of input–output data and leverage it to design model-free RL algorithms. In addition, we study the statistic of the estimations. Note that we neither favor the approach of feeding a finite history of input–output data for partially observable dynamical systems, nor do we deny its importance. We aim to quantify the effect of this approach, and this is the contribution ofthis article. Compared with [17], first we consider stochastic systems and take into account the process and observation noises and second, study and quantify the effect of noise on the average cost when a batch of input–output data is utilized.

Manuscript received 15 August 2023; revised 21 August 2023 and 5 February 2024; accepted 25 March 2024. Date of publication 5 April 2024; date of current version 29 August 2024. This work was supported in part by the Wallenberg AI, Autonomous Systems and Software Program (WASP) through the Knut and Alice Wallenberg Foundation. The work of Farnaz Adib Yaghmaie was supported in part by the ZENITH, Excellence Center at Linköping–Lund in Information Technology (ELLIIT), in part by the Sensor informatics and Decision-making for the Digital Transformation (SEDDIT), and in part by the Wallenberg AI, Autonomous Systems and Software Program (WASP) through the Knut and Alice Wallenberg Foundation. The work of Hamidreza Modares was supported by the National Science Foundation under grant ECCS-2227311. The work of Fredrik Gustafsson was supported by the Vinnova Competence Center LINK-SIC and the Scalable Kalman Filters project through the Swedish Research Council. Recommended by Associate Editor F. Zhang. *(Corresponding author: Hamidreza Modares.)*

Farnaz Adib Yaghmaie and Fredrik Gustafsson are with the Faculty of Electrical Engineering, Linköping University, 58183 Linköping, Sweden (e-mail: farnaz.adib.yaghmaie@liu.se; fredrik.gustafsson@liu.se).

Hamidreza Modares is with the College of Engineering, Michigan State University, East Lansing, MI 48824 USA (e-mail: modaresh@msu.edu).

Digital Object Identifier 10.1109/TAC.2024.3385680

II. LQ Problem

Notations: Let $\mathbb{R}^{n\times m}$ denote the set of real $n\times m$ matrices. $\mathbf{0}$ denotes a matrix with appropriate dimensions and zero entries. For $A\in\mathbb{R}^{n\times n}$, $\rho(A)$ and $\mathrm{Tr}(A)$ denote the spectral radius and the trace of A, respectively. A (semi)positive definite matrix A is denoted by $A>0$ ($A\geq 0$). Let A denote a matrix. Then, $\mathrm{Diag}_L\{A\}$ builds a block diagonal matrix where the matrix A is repeated L times on the diagonal blocks. Let a_k denote a vector at time k. Define $a_{k:L}:=\begin{bmatrix}a_k^T & a_{k+1}^T & \dots & a_{k+L-1}^T\end{bmatrix}^T$.

A. Linear Gaussian Dynamical System

Consider a linear Gaussian dynamical system

$$\begin{aligned} x_{k+1}&=Ax_k+Bu_k+Gw_k\\ y_k&=Cx_k+v_k \end{aligned} \tag{1}$$

where $x_k\in\mathbb{R}^n$, $u_k\in\mathbb{R}^m$, and $y_k\in\mathbb{R}^p$ are the state, the input, and the output vectors, respectively. $w_k\in\mathbb{R}^q$ and $v_k\in\mathbb{R}^p$ denote the process and observation noises drawn independent identically distributed from Gaussian distributions $w_k\sim\mathcal{N}(\mathbf{0},W_w)$, $v_k\sim\mathcal{N}(\mathbf{0},W_v)$, and $\mathbb{E}[w_kv_j^T]=\mathbf{0}$. Since the state variable x_k is not directly measurable, we assume that an estimator exists and returns the following estimation:

$$\hat{x}_k=x_k+n_k \tag{2}$$

where $n_k\sim\mathcal{N}(\mathbf{0},W_n)$ denotes the noise in the estimation. $\hat{x}_k$ can be any unbiased estimator of x_k.

Assumption 1: The pair (A,B) is controllable and the pair (A,C) is observable.

Assumption 2: n_k is independent of the w_k; that is, $\mathbb{E}[w_kn_k^T]=\mathbf{0}$ but can be correlated with v_k, w_{k-j}, v_{k-j}, $j=1,\dots,k-1$. An important implication of this assumption is that $\mathbb{E}[x_kn_k^T]=W_{xn}\neq\mathbf{0}$ in general.

This assumption can be made without loss of generality. Indeed, if the state variable is known exactly $n_k\equiv\mathbf{0}$, and we can set the covariance matrices related to n_k equal to zero.

Using the estimated state variable (2), we are interested in designing a state feedback for the system in (1)

$$u_k=K\hat{x}_k=K(x_k+n_k) \tag{3}$$

where K is a stabilizing controller gain, $\rho(A+BK)<1$, and can be designed optimally.

Let $x_0\sim\mathcal{N}(\mathbf{0},X_0)$ and $x_\infty\to\mathcal{N}(m_\infty,X_\infty)$ denote the initial and the stationary state distribution, respectively. If K is stabilizing, it is easy to verify that $m_\infty=\mathbf{0}$. Using (3) in (1), one can obtain the stationary distribution of x_k

$$\begin{aligned} \mathbb{E}[x_{k+1}x_{k+1}^T]&=(A+BK)\mathbb{E}[x_kx_k^T](A+BK)^T+GW_wG^T\\ &\quad+BKW_nK^TB^T+(A+BK)\underbrace{\mathbb{E}[x_kw_k^T]}_{=\mathbf{0}}G^T\\ &\quad+G\underbrace{\mathbb{E}[w_kx_k^T]}_{=\mathbf{0}}(A+BK)^T\\ &\quad+(A+BK)\mathbb{E}[x_kn_k^T]K^TB^T\\ &\quad+BK\mathbb{E}[n_kx_k^T](A+BK)^T+G\underbrace{\mathbb{E}[w_kn_k^T]}_{=\mathbf{0}}K^TB^T\\ &\quad+BK\underbrace{\mathbb{E}[n_kw_k^T]}_{=\mathbf{0}}G^T. \end{aligned}$$

Using $\mathbb{E}[x_{k+1}x_{k+1}^T]=\mathbb{E}[x_kx_k^T]=X_\infty$, we get

$$\begin{aligned} X_\infty&=(A+BK)X_\infty(A+BK)^T\\ &\quad+GW_wG^T+BKW_nK^TB^T\\ &\quad+(A+BK)W_{xn}K^TB^T+BKW_{xn}^T(A+BK)^T. \end{aligned} \tag{4}$$

B. Average Cost

Define a quadratic running cost for (1)

$$r_k(y_k,u_k)=y_k^TR_yy_k+u_k^TR_uu_k \tag{5}$$

where $R_y\geq 0$ and $R_u>0$. In this article, we consider an average cost as the performance index

$$\lambda=\lim_{\tau\to\infty}\frac{1}{\tau}\mathbb{E}\left[\sum_{t=1}^{\tau}r_t(y_t,u_t)\right]. \tag{6}$$

Lemma 1: Consider the dynamical system in (1) and let Assumptions 1 and 2 hold. Assume that the gain K is stabilizing. The average cost associated with the policy in (3) is

$$\begin{aligned} \lambda(K)&=\mathrm{Tr}(C^TR_yCX_\infty+K^TR_uKX_\infty)+\mathrm{Tr}(R_yW_v)\\ &\quad+\mathrm{Tr}(K^TR_uKW_n)+2\mathrm{Tr}(K^TR_uKW_{xn}). \end{aligned} \tag{7}$$

Proof: See Appendix A. ■

C. Optimal and Nonoptimal Average Costs

Problem 1 (Linear quadratic Gaussian (LQG) problem [9], [18], [19]): Consider the dynamical system in (1) and assume that (A,B,C), and W_w and W_v are known. Design the observer $\hat{x}_k$ and the controller gain K^* such that $u_k=K^*\hat{x}_k$ minimizes the average cost (7).

The LQG Problem 1 is to estimate the state variable via a Kalman filter and use the estimated state as if it were the actual state in the controller. The solution to Problem 1 exists under Assumption 1 and is given by ([9], [18], [19])

$$\begin{aligned} u_k&=K^*\hat{x}_{k|k}\\ \hat{x}_{k+1|k+1}Y&=A\hat{x}_{k|k}+Bu_k+F^*(y_{k+1}-C(A\hat{x}_{k|k}+Bu_k)) \end{aligned} \tag{8}$$

where $\hat{x}_{k|k}$ is a posteriori state estimation at time k given observations $(y_1,\dots,y_k)$, F^* is the observer gain

$$\begin{aligned} &AM^*A^T-M^*-AM^*C^T(CM^*C^T+W_v)^{-1}CM^*A^T\\ &\quad+GW_wG^T=\mathbf{0}\\ &F^*=-M^*C^T(W_v+CM^*C^T)^{-1} \end{aligned} \tag{9}$$

and K^* is the controller gain derived from the algebraic Riccati equation (ARE)

$$\begin{aligned} &A^TP^*A-P^*-A^TP^*B(B^TP^*B+R_u)^{-1}B^TP^*A\\ &\quad+C^TR_yC=\mathbf{0} \end{aligned} \tag{10}$$

$$K^*=-(R_u+B^TP^*B)^{-1}B^TP^*A. \tag{11}$$

The ARE in (10) can also be written as

$$\begin{aligned} &(A+BK^*)^TP^*(A+BK^*)-P^*+K^{*T}R_uK^*\\ &\quad+C^TR_yC=\mathbf{0}. \end{aligned} \tag{12}$$

The optimal average cost for the LQG can be divided into two parts, (see [18, section 9.3.2] or [20, section 5.2] for similar discussions)

$$\lambda^*=\lambda_{\mathrm{lqr}}+\lambda_{\mathrm{est}} \tag{13}$$

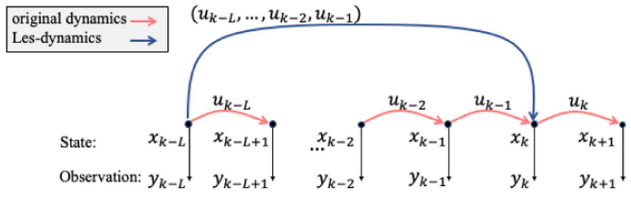

Fig. 1. Les-dynamics.

where the following holds.

1) λ_{lqr} is the cost if we knew the state. To obtain λ_{lqr}, use K^* in (7) and set $W_n = \mathbf{0}$ and $W_{xn} = \mathbf{0}$

$$\begin{aligned}\lambda_{\text{lqr}} &= \text{Tr}(C^T R_y C X_\infty + K^{*T} R_u K^* X_\infty) + \text{Tr}(R_y W_v) \\ &= \text{Tr}(P^* X_\infty - (A+BK^*)^T P^* (A+BK^*) X_\infty) \\ &\quad + \text{Tr}(R_y W_v) \quad \text{using (12)} \\ &= \text{Tr}(P^* G W_w G^T) + \text{Tr}(R_y W_v) \text{ using (4).} \end{aligned} \tag{14}$$

2) λ_{est} is the cost of estimating the state

$$\begin{aligned}\lambda_{\text{est}} &= \text{Tr}((C^T R_y C - P^*)\Sigma) + \text{Tr}(A^T P^* A \Sigma) \\ &= \text{Tr}(A^T P^* B (B^T P^* B + R_u)^{-1} B^T P^* A \Sigma) \end{aligned} \tag{15}$$

where

$$\Sigma = M^* - M^* C^T (C M^* C^T + W_v)^{-1} C M^* \tag{16}$$

and M^* is given in (9).

The average cost given in (13) is the minimum average cost for the dynamical system (1). Any other controller gain $K \neq K^*$ or other means of estimating x_k will result in an increase in the average cost $\lambda(K) > \lambda^*$. An example is when a finite history of the input–output data is fed instead of the state variable. In the next section, we show that using a finite history of input–output data, we can define a point estimator for the state variable. Since this estimator is not a Kalman filter (and as such, not an optimal estimator), this approach results in more average cost.

III. Les-Dynamics

In this article, we coin the new concept of *Les-dynamics*, which is an equivalent representation of the original dynamics (1) by collecting L extra samplings and captures the dynamics of the system over L steps; see Fig. 1. Note that it is not required to sample faster; we only need to collect L samples at the normal sampling rate of the system. Using Les-dynamics, we can define an estimator for x_k based on the past inputs and outputs and show that $\hat{x}_k = x_k + n_k$ in (2) holds.

Theorem 1: Let Assumption 1 be satisfied. By collecting L extra samples, the linear model (1) is equivalent to the *Les-dynamics*

$$\begin{aligned}\bar{x}_l &= A_L \bar{x}_{l-1} + B_L \bar{u}_{l-1} + G_L \bar{w}_{l-1} \\ \bar{y}_{l-1} &= C_L \bar{x}_{l-1} + D_L \bar{u}_{l-1} + H_L \bar{w}_{l-1} + \bar{v}_{l-1} \end{aligned} \tag{17}$$

where

$$A_L = [A^L], \quad B_L = \begin{bmatrix} A^{L-1}B & \ldots & AB & B \end{bmatrix}$$

$$G_L = \begin{bmatrix} A^{L-1}G & \ldots & AG & G \end{bmatrix}$$

$$D_L = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} \\ CB & \mathbf{0} & \ldots & \vdots \\ CAB & \ddots & & \\ \vdots & & CB & \mathbf{0} \\ CA^{L-1}B & \ldots & CAB & CB \end{bmatrix}, \quad C_L = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{L-1} \\ CA^L \end{bmatrix}$$

$$H_L = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \ldots & \mathbf{0} \\ CG & \mathbf{0} & \ldots & \vdots \\ CAG & \ddots & & \\ \vdots & & CG & \mathbf{0} \\ CA^{L-1}G & \ldots & CAG & CG \end{bmatrix}, \quad W_L = \begin{bmatrix} W_{wL} & \mathbf{0} \\ \mathbf{0} & W_{vL} \end{bmatrix}$$

$$\begin{aligned}&\text{Cov}(w_{l-1}) = W_{wL} = [\text{Diag}_L\{W_w\}] \\ &\text{Cov}(v_{l-1}) = W_{vL} = [\text{Diag}_{L+1}\{W_v\}] \end{aligned} \tag{18}$$

$$\begin{aligned}&\bar{x}_{l-1} := x_{k-L}, \ \bar{x}_l := x_k \\ &\bar{u}_{l-1} := u_{k-L:L}, \ \bar{y}_{l-1} := y_{k-L:L+1} \\ &\bar{w}_{l-1} := w_{k-L:L}, \ \bar{v}_{l-1} := v_{k-L:L+1}. \end{aligned} \tag{19}$$

Proof: See Appendix B. ■

Assumption 3: Define $C_0 := C$, and assume that $L \geq 0$ is the smallest integer such that C_L in (18) has full column rank. Note that L exists under Assumption 1.

A. Point Estimator of the Current State x_k

Using the concept of Les-dynamics, we define a point estimator of the state x_k in Theorem 2. Let

$$\begin{aligned}&Y := A_L C_L^+ H_L - G_L = [Y_t], \ Y_t \in \mathbb{R}^{n \times q}, \quad t = 1, \ldots, L \\ &Z := A_L C_L^+ = [Z_t], \ Z_t \in \mathbb{R}^{n \times p}, \quad t = 1, \ldots, L, L+1 \end{aligned} \tag{20}$$

where

$$\begin{aligned}C_L^+ &= (C_L^T W_{eL}^{-1} C_L)^{-1} C_L^T W_{eL}^{-1} \\ W_{eL} &= \begin{bmatrix} H_L & I_{(L+1)p} \end{bmatrix} W_L \begin{bmatrix} H_L^T \\ I_{(L+1)p} \end{bmatrix}. \end{aligned} \tag{21}$$

Note that C_L^+ is a left inverse of C_L; i.e., $C_L^+ C_L = I_n$.

Theorem 2: Consider the dynamical system in (1) and the Les-dynamics in (17). Let Assumptions 1 and 3 hold. There exists a maximum likelihood estimator of x_k such that $\hat{x}_k = x_k + n_k$, where $\hat{x}_k$ is given by

$$\begin{aligned}\hat{x}_k &= A_L C_L^+ \bar{y}_{l-1} + (B_L - A_L C_L^+ D_L) \bar{u}_{l-1} \\ &= \Gamma [\bar{y}_{l-1}^T, \ \bar{u}_{l-1}^T]^T \end{aligned} \tag{22}$$

and n_k is the noise in the estimation

$$n_k = Y \bar{w}_{l-1} + Z \bar{v}_{l-1} \tag{23}$$

$$\mathbb{E}[n_k n_k^T] = W_n = \text{Cov}(\hat{x}_k) = Y W_{wL} Y^T + Z W_{vL} Z^T. \tag{24}$$

Proof: See Appendix C. ■

B. Properties of the Point Estimator of x_k in (22)

Theorem 3: Consider the dynamical system in (1) and the Les-dynamics in (17). Let Assumptions 1 and 3 hold. Let $\hat{x}_k|_L$ and

$\text{Cov}(\hat{x}_k)|_L$ denote the estimate of x_k and the covariance of estimation using data of length L, respectively. For $L_1 > L$, where L satisfies Assumption 3

$$\hat{x}_k|_{L_1} = \hat{x}_k|_L,\ \text{Cov}(\hat{x}_k)|_{L_1} = \text{Cov}(\hat{x}_k)|_L. \tag{25}$$

Proof: See Appendix D. ■

Remark 1: By Theorem 3, using a longer history of data does not help us to have a better estimation of the current state x_k. As a result, it is better to keep the history length to the minimum where C_L has full column rank.

The noise n_k in (23) is correlated with the state x_k because it depends on w_{k-i}, $i < k$, and w_{k-i} contributes to x_k. We quantify the correlation in the next lemma.

Lemma 2: Consider (1), and let Assumptions 1 and 3 hold. Assume that the policy $u_k = K\hat{x}_k = K(x_k + n_k)$ in (3) is used to control the system, where $\hat{x}_k$ and n_k are given in (22) and (23). Then, for $k \geq L+1$

$$\begin{aligned}\mathbb{E}[x_k n_k^T] = W_{xn} = &\sum_{j=1}^{L-1}(A+BK)^{j-1}BK\sum_{t=1}^{L-j}Y_{t+j}W_wY_t^T\\ &+\sum_{j=1}^{L-1}(A+BK)^{j-1}BK\sum_{t=1}^{L-j+1}Z_{t+j}W_vZ_t^T\\ &+(A+BK)^{L-1}BKZ_LW_vZ_1^T\\ &+\sum_{j=1}^{L}(A+BK)^{L-j}GW_wY_j^T.\end{aligned} \tag{26}$$

Note that Y_t and Z_t are defined in (20).

Proof: See Appendix E. ■

Remark 2: One can use history of input–output data to define state estimator $\hat{x}_k$ in (22). The average cost associated with selecting the controller as $u_k = K\hat{x}_k$ is given by (7), where W_n and W_{xn} are given in (24) and (26), respectively, while the optimal average cost is obtained by using a Kalman filter in the LQG controller and it is specified in (13).

C. Incorporating Estimator in RL Framework

The estimator in (22) depends on the system dynamics yet our first assumption in RL is that the dynamics is unknown. However, we can push the estimator Γ to the structure that is learned.

Learning the policy: In policy search algorithms, a policy is learned directly. When the state variable is not available and we feed $[\bar{y}_{l-1}^T,\ \bar{u}_{l-1}^T]^T$ to a policy search algorithm, we can write the controller as $u = K\Gamma[\bar{y}_{l-1}^T,\ \bar{u}_{l-1}^T]^T$ and the algorithm will learn $K\Gamma$.

Learning value function: In dynamic programming-based algorithms, a value function is learned. The state–value function associated with the policy $u_k = Kx_k$ is defined as

$$V(x_k, K) = \mathbf{E}\left[\sum_{t=k}^{+\infty}(r_k - \lambda(K))|x_k\right]. \tag{27}$$

It has been shown in [10] and [21] that the value function is quadratic in x_k, $V(x_k) = x_k^TPx_k$, and the dynamic programming-based algorithms learn the kernel P. When the state variable is not available, we can write

$$V(\hat{x}_k, K) = \hat{x}_k^TP\hat{x}_k = \begin{bmatrix}\bar{y}_{l-1}^T & \bar{u}_{l-1}^T\end{bmatrix}\Gamma^TP\Gamma\begin{bmatrix}\bar{y}_{l-1}\\ \bar{u}_{l-1}\end{bmatrix}.$$

As a result, the algorithm will learn $\Gamma^TP\Gamma$. Extensions to the Q-function and advantage function are similar.

IV. Simulation Results

Consider the following problem setup:

$$x_{k+1} = \begin{bmatrix}0 & 1\\ 0.45 & -0.4\end{bmatrix}x_k + \begin{bmatrix}0\\ 1\end{bmatrix}u_k + w_k,\quad x_0 \sim \mathcal{N}(\mathbf{0}, I_2)$$

$$y_k = \begin{bmatrix}1 & 0\end{bmatrix}x_k + v_k$$

$$W_w = I_2,\ W_v = 1,\ R_y = 1,\ R_u = 1.$$

For $L = 1$, C_L in (18) has full column rank meaning that x_k can be estimated from $[y_k, y_{k-1}, u_{k-1}]$ in each time-step.

A. Optimal Solution Using Full Information of the Dynamics

Our baseline for comparison is the optimal analytical solution, discussed in Section II-C, assuming that the dynamics (A, B, C) is exactly known. The solutions to (9)–(11) are

$$P^* = \begin{bmatrix}1.1120 & -0.1246\\ -0.1246 & 1.2381\end{bmatrix},\ K^* = \begin{bmatrix}-0.2485 & 0.2770\end{bmatrix}$$

$$M^* = \begin{bmatrix}2.3054 & -0.6045\\ -0.6045 & 1.4159\end{bmatrix},\ F^* = \begin{bmatrix}0.6975\\ -0.1829\end{bmatrix}.$$

Then, by (13)–(15), $\lambda_{\text{lqr}} = 3.35017$ and $\lambda_{\text{est}} = 0.37730$. The optimal average cost is $\lambda^* = \lambda_{\text{lqr}} + \lambda_{\text{est}} = 3.72747$.

B. Evaluated Algorithms

We select the algorithms from the dynamic programming-based, policy search, and model building approaches. The initial controller for all algorithms is selected as $K^0[y_k, y_{k-1}, u_{k-1}]^T$ where the entries of K^0 are selected from a standard distribution and multiplied by 0.01 to be near zero since A is stable.

1) Dynamic Programming-Based RL: Each algorithm iterates n_{dp} times and in each iteration, a rollout of length T is collected.

Off-policy learning [10]*:* The behavioral policy is selected as $u_k = K^i[y_k,\ y_{k-1},\ u_{k-1}]^T + \eta_k$, where $\eta_k \sim \mathcal{N}(\mathbf{0}, 1)$.

Q-learning [22]*:* The behavioral policy is selected as $u_k = K^i[y_k,\ y_{k-1},\ u_{k-1}]^T + \eta_k$, where $\eta_k \sim \mathcal{N}(\mathbf{0}, 1)$.

Average off-policy learning [11]*:* We set $\tau' = T$ and select $\tau'' = 0$. The behavioral policy is selected as $u_k = K^i[y_k,\ y_{k-1},\ u_{k-1}]^T + \eta_k$, where $\eta_k \sim \mathcal{N}(\mathbf{0}, 1)$.

2) Stochastic Policy Gradient [1]: The algorithm iterates n_{pg} times and in each iteration, n_{batch} mini-batches of length T_{batch} are collected. The probability density function of the probabilistic policy is assumed to be Gaussian $\mathcal{N}(K^i[y_k,\ y_{k-1},\ u_{k-1}]^T, 0.01I_3)$. We use Adam optimizer to update the controller gain with 0.1 as the learning rate and default values for other hyperparameters.

3) Model-Building RL: We collect T_{mb} input–output samples by applying $u_k = K^0[y_k, y_{k-1}, u_{k-1}]^T$ on the system. We use subspace methods to estimate the dynamics (A, B, C), see [7, Section 7.3]. We use the estimated dynamics $(\hat{A}, \hat{B}, \hat{C})$ in (9)–(11), and the controller gain K_{mb} and the Kalman filter gain F_{mb} accordingly. The controller $u_k = K_{\text{mb}}\hat{x}_{k|k}$, where $\hat{x}_{k|k}$ is the state estimation by the Kalman filer in (8) is applied to the original system for the evaluation.

C. Performance Evaluation

We evaluate the performance of the algorithms against the number of samples. We set the algorithms' parameters including the number of iterations, rollout length, and batch length such that all algorithms have the same sample budget. For the dynamic programming-based algorithms, we set $n_{\text{dp}} = 3$ and change the rollout length

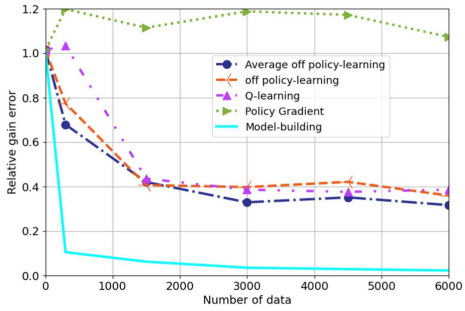

Fig. 2. Median of error in the learned controller gains normalized with $K^*\Gamma$.

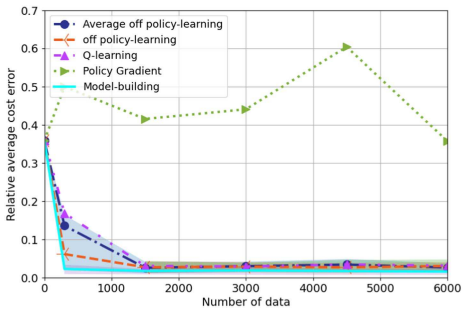

Fig. 3. Mean of $\frac{|\lambda(K)-\lambda^*|}{\lambda^*}$, where K is the controller gain generated by the algorithms. Shaded regions display quartiles.

as $T = [100,\ 500,\ 1000,\ 15000]$. In the policy gradient, we set $T_{\text{batch}} = 10$, $n_{\text{batch}} = 6$, and change the number of iterations as $n_{\text{pg}} = (n_{\text{dp}}/(T_{\text{batch}} n_{\text{batch}}))T$. In the model-building approach $T_{\text{mb}} = n_{\text{dp}} T = [300,\ 1500,\ 3000,\ 45\,000]$. We run each algorithm 50 times.

In Fig. 2, we plot the median of the relative error between the learned controller gain by the algorithms and the optimal analytical gain. Note that since the evaluated algorithms use $[y_k,\ y_{k-1},\ u_{k-1}]$ and learn $K\Gamma$, we need to compare with $K^*\Gamma$. The controller gain by the policy gradient is far away from the analytical optimal gain because the policy gradient algorithm tries to directly optimize the performance index and as such, it is sensitive to observation noise. The controller gains by the dynamic-programming algorithms become closer to $K^*\Gamma$ as the number of data increases. The controller by the model-building approach is the closest to the optimal solution.

We use the generated controller gains by the algorithms to run the linear Gaussian system for 5000 steps and compute the average cost empirically. In Fig. 3, we report the mean of the relative average cost $\frac{|\lambda(K)-\lambda^*|}{\lambda^*}$, where $\lambda^* = 3.72747$ is the optimal analytical average cost using the full knowledge of the dynamics discussed in Section IV-A, and $\lambda(K)$ is the average cost when the controller is designed by one of the aforementioned RL algorithms. The performance of the stochastic policy gradient algorithm is OFF from the optimal behavior. The reason is that policy search algorithms are sensitive to noise and the estimator noise n_k in (23) degenerates the performance of the stochastic policy gradient algorithm during sampling actions and calculating the performance gradient. The average cost by the model-building approach is close to the analytical optimal average cost, and it is superb as it gets a good performance by only using 300 data points. Dynamic programming-based algorithms have also good performances, and they are close to the analytical optimal average cost without knowing the dynamics.

Remark 3: The effects of the process and observation noises in the estimation of the value functions have been extensively studied in [10] and [21]. Given a good estimation of the value function, the convergence of the policy iteration algorithms is discussed in [10] and [23]. The convergence of policy gradient algorithms for linear systems with process noise is studied in [5] and [24], and the effect of observation noise on the convergence has been numerically shown in [10].

V. Conclusion

In this article, we have considered linear Gaussian dynamical systems where the state variable is only partially observable and noisy. We have assumed that there exists a state estimator that returns unbiased but noisy estimations of the state vector. We have quantified the effect of using estimation of the state in the average cost. We have further discussed that the minimum average cost can be achieved if a Kalman filter, which is the optimal state estimator, is used. Later, we have introduced the new concept of Les-dynamics and defined a Les-dynamics state estimator: the idea is to batch the dynamics of the system and use a finite history of the input–output data to define the state estimator. There are pros and cons with a Les-dynamics estimator: a Les-dynamics estimator is not optimal (because it is not a Kalman filter) and as such, it results in a higher average cost; however, the advantage is that the structure of the Les-dynamics estimator can be incorporated into the RL framework. We have also shown that it is better to keep the history length to a minimum to avoid unnecessary parameter estimation.

If the dynamic is unknown, one can follow two alternative paths. 1) To estimate the dynamics and then use a Kalman filter as the state estimator using the estimated dynamics. This approach is called "model building" in the simulation results. 2) To use a finite history of input–output data in model-free RL approaches. None of these approaches return the optimal average cost. In the model-building approach, the dynamic is estimated and is not exact. So, the analytical optimal average cost cannot be achieved. The state estimate by a finite history of input–output data will never be as good as using a Kalman filter with exact knowledge of the dynamics.

We have shown in our empirical evaluation that the average costs of the model-building approach and dynamic-programming based RL algorithms are close to each other and also close to the optimal analytical average cost. This shows that using a finite history of input–output data is a powerful approach in dynamic programming-based and model-building RL approaches. However, using a finite history of input–output data does not produce good results in the policy gradient algorithm because policy search algorithms are sensitive to noise. The statistics of the presented state estimator and its effect on the cost are provided, which opens up a future research direction to design rick-informed controllers that not only account for the average cost but also its variance.

Appendix A
Proof of Lemma 1

We derive the average cost by directly inserting y_k and $u_k = K\hat{x}_k = K(x_k + n_k)$ in (6)

$$\lambda(K) = \lim_{\tau\to\infty} \frac{1}{\tau}\mathbb{E}\left[\sum_{t=1}^{\tau}(x_t^T C^T + v_t^T)R_y(Cx_t + v_t)\right]$$
$$+ \lim_{\tau\to\infty} \frac{1}{\tau}\mathbb{E}\left[\sum_{t=1}^{\tau}(x_t + n_t)^T K^T R_u K(x_t + n_t)\right]$$

$$= \lim_{\tau\to\infty}\frac{1}{\tau}\sum_{t=1}^{\tau}\mathbb{E}[x_t^T C^T R_y C x_t + 2v_t^T R_y C x_t + v_t^T R_y v_t]$$
$$+ \lim_{\tau\to\infty}\frac{1}{\tau}\sum_{t=1}^{\tau}\mathbb{E}[x_t^T K^T R_u K x_t + 2n_t^T K^T R_u K x_t]$$
$$+ \lim_{\tau\to\infty}\frac{1}{\tau}\sum_{t=1}^{\tau}\mathbb{E}[n_t^T K^T R_u K n_t].$$

Using $\mathbb{E}[x_t^T v_t] = 0,\ \mathrm{Tr}(AB) = \mathrm{Tr}(BA)$

$$\lambda(K) = \lim_{\tau\to\infty}\frac{1}{\tau}\sum_{t=1}^{\tau}[\mathrm{Tr}(C^T R_y C\mathbb{E}[x_t x_t^T]) + \mathrm{Tr}(R_y\mathbb{E}[v_t v_t^T])]$$
$$+ \lim_{\tau\to\infty}\frac{1}{\tau}\sum_{t=1}^{\tau}[\mathrm{Tr}(K^T R_u K\mathbb{E}[x_t x_t^T])$$
$$+ \mathrm{Tr}(2\,K^T R_u K\mathbb{E}[x_t n_t^T])]$$
$$+ \lim_{\tau\to\infty}\frac{1}{\tau}\sum_{t=1}^{\tau}\mathrm{Tr}(K^T R_u K\mathbb{E}[n_t n_t^T]).$$

Using $\mathbb{E}[x_t x_t^T] = X_\infty,\ \mathbb{E}[v_t v_t^T]] = W_v,\ \mathbb{E}[x_t n_t^T] = W_{xn}$, and $\mathbb{E}[n_t n_t^T] = W_n$, one gets

$$\lambda(K) = \lim_{\tau\to\infty}\frac{1}{\tau}\tau\mathrm{Tr}(C^T R_y C X_\infty + K^T R_u K X_\infty)$$
$$+ \lim_{\tau\to\infty}\frac{1}{\tau}[\tau\mathrm{Tr}(R_y W_v) + \tau\mathrm{Tr}(2\,K^T R_u K W_{xn})]$$
$$+ \lim_{\tau\to\infty}\frac{1}{\tau}\tau\mathrm{Tr}(K^T R_u K W_n)$$
$$= \mathrm{Tr}(C^T R_y C X_\infty + K^T R_u K X_\infty) + \mathrm{Tr}(R_y W_v)$$
$$+ 2\mathrm{Tr}(K^T R_u K W_{xn}) + \mathrm{Tr}(K^T R_u K W_n).$$

Appendix B
Proof of Theorem 1

Collecting one extra output sampling at $k+1$ from the system in (1)

$$x_{k+1} = Ax_k + Bu_k + Gw_k$$
$$y_{k+1} = CAx_k + CBu_k + CGw_k + v_{k+1}.$$

We can continue like this and collect L extra samples. By stacking $L+1$ adjacent measurements from y_k to y_{k+L} and building $y_{k:L+1}$, we have

$$x_{k+L} = A_L x_k + B_L u_{k:L} + G_L w_{k:L}$$
$$y_{k:L+1} = C_L x_k + D_L u_{k:L} + H_L w_{k:L} + v_{k:L+1}$$

where $A_L,\ B_L,\ G_L,\ C_L,\ D_L,$ and H_L are defined in (18). By shifting the time index from k to $k-L$, the dynamics is defined over the horizon $[k-L,\ k]$

$$x_k = A_L x_{k-L} + B_L u_{k-L:L} + G_L w_{k-L:L}$$
$$y_{k-L:L+1} = C_L x_{k-L} + D_L u_{k-L:L} + H_L w_{k-L:L} + v_{k-L:L+1}.$$

Let $\bar{x}_{l-1} := x_{k-L},\ \bar{x}_l := x_k$, and (17) is concluded.

Appendix C
Proof of Theorem 2

Since C_L has full column rank by Assumption 3, there exists a matrix $N_L \in \mathbb{R}^{n\times Lp}$ such that $A_L = N_L C_L$ and $N_L = A_L C_L^+$. Then, we have

$$A_L\bar{x}_{l-1} = N_L C_L \bar{x}_{l-1}$$
$$= N_L(\bar{y}_{l-1} - D_L\bar{u}_{l-1} - H_L\bar{w}_{l-1} - \bar{v}_{l-1})$$

where we have substituted $C_L\bar{x}_{l-1}$ from the output equation in (17). Substituting the above result in the state equation in (17), $x_k \equiv \bar{x}_l$ reads

$$x_k = N_L\bar{y}_{l-1} + (B_L - N_L D_L)\bar{u}_{l-1} + (G_L - N_L H_L)\bar{w}_{l-1} - N_L\bar{v}_{l-1}. \tag{28}$$

Hence, the maximum likelihood estimation of $\bar{x}_l$ is given by (22), and the noise n_k in the estimation is given by (23).

Appendix D
Proof of Theorem 3

For simplicity, we select $L_1 = L+1$. The proof can be extended for general $L_1 > L$ easily. Let $\mathcal{D}_L$ represent the collection of input, output, process, and observation noises over the interval $[k-L, k)$

$$\mathcal{D}_{k-L:L} = \{\bar{u}_l,\ \bar{y}_l,\ \bar{w}_l,\ \bar{v}_l\}.$$

Let $\mathcal{D}_n = \{u_n,\ y_n,\ w_n,\ v_n\}$ denote the new data point, which has happened before $\mathcal{D}_{k-L:L}$. $\mathcal{D}_{k-L-1:L+1}$ reads

$$\mathcal{D}_{k-L-1:L+1} = \{[u_n,\ \bar{u}_{l-1}],\ [y_n,\ \bar{y}_{l-1}],\ [w_n,\ \bar{w}_{l-1}],\ [v_n,\ \bar{v}_{l-1}]\}.$$

According to (18), the following identities are concluded immediately:

$$A_{L+1} = A^{L+1},\ B_{L+1} = \begin{bmatrix}A^L B & B_L\end{bmatrix},\ G_{L+1} = \begin{bmatrix}A^L G & G_L\end{bmatrix}$$
$$C_{L+1} = \begin{bmatrix}C\\ C_L A\end{bmatrix},\ D_{L+1} = \begin{bmatrix}\mathbf{0} & \mathbf{0}\\ C_L B & D_L\end{bmatrix}$$
$$H_{L+1} = \begin{bmatrix}\mathbf{0} & \mathbf{0}\\ C_L G & H_L\end{bmatrix}. \tag{29}$$

We can also show that $N_{L+1} = \begin{bmatrix}\mathbf{0} & A_L C_L^+\end{bmatrix}$. Let $N_{L+1} = [n_1, N_L]$. According to the definition of N_L in the proof of Theorem 2, one has $A_{L+1} = N_{L+1}C_{L+1}$. Replacing A_{L+1} and C_{L+1} from (29), one have

$$A^{L+1} = [n_1, N_L]\begin{bmatrix}C\\ C_L A\end{bmatrix} = n_1 C + N_L C_L A.$$

Since $N_L C_L = A_L = A^L$, one gets $A^{L+1} = n_1 C + A^{L+1}$ and as a result, one can select $n_1 = \mathbf{0}$. Now we prove the theorem. We first show that $\hat{x}_k|_{L_1} = \hat{x}_k|_L$

$$\hat{x}_k|_{L_1} = A_{L+1}C_{L+1}^+\begin{bmatrix}y_n\\ \bar{y}_{l-1}\end{bmatrix}$$
$$+ (B_{L+1} - A_{L+1}C_{L+1}^+ D_{L+1})\begin{bmatrix}u_n\\ \bar{u}_{l-1}\end{bmatrix}$$
$$= \begin{bmatrix}\mathbf{0} & A_L C_L^+\end{bmatrix}\begin{bmatrix}y_n\\ \bar{y}_{l-1}\end{bmatrix}$$

$$+\left(\begin{bmatrix}A^LB & B_L\end{bmatrix}-\begin{bmatrix}\mathbf{0} & N_L\end{bmatrix}\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ C_LB & D_L\end{bmatrix}\right)\begin{bmatrix}u_n\\ \bar{u}_{l-1}\end{bmatrix}$$
$$=N_L\bar{y}_{l-1}+\begin{bmatrix}A^LB-N_LC_LB & B_L-N_LD_L\end{bmatrix}\begin{bmatrix}u_n\\ \bar{u}_{l-1}\end{bmatrix}$$
$$=A_LC_L^+\bar{y}_{l-1}+(B_L-A_LC_L^+D_L)\bar{u}_{l-1}=\hat{x}_k|_L.$$

Second, we prove $\mathrm{Cov}(\hat{x}_k)|_{L_1}=\mathrm{Cov}(\hat{x}_k)|_L$. Let $n_k|_L$ denote the noise in the estimation using data of length L. We prove that $n_k|_{L_1}\equiv n_k|_L$

$$n_k|_{L_1}=(A_{L+1}C_{L+1}^+H_{L+1}-G_{L+1})\begin{bmatrix}w_n\\ \bar{w}_{l-1}\end{bmatrix}$$
$$+A_{L+1}C_{L+1}^+\begin{bmatrix}v_n\\ \bar{v}_{l-1}\end{bmatrix}$$
$$=\left(\begin{bmatrix}\mathbf{0} & N_L\end{bmatrix}\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ C_LG & H_L\end{bmatrix}-\begin{bmatrix}A^LG & G_L\end{bmatrix}\right)\begin{bmatrix}w_n\\ \bar{w}_{l-1}\end{bmatrix}$$
$$+\begin{bmatrix}\mathbf{0} & N_L\end{bmatrix}\begin{bmatrix}v_n\\ \bar{v}_{l-1}\end{bmatrix}$$
$$=(A_LC_L^+H_L-G_L)\bar{w}_{l-1}+A_LC_L^+\bar{v}_{l-1}=n_k|_L.$$

Appendix E
Proof of Lemma 2

The solution to (1) using (3) reads

$$x_k=(A+BK)^kx_0+\sum_{j=1}^{k-1}(A+BK)^{j-1}BKn_{k-j}$$
$$+\sum_{j=1}^{k-1}(A+BK)^{j-1}Gw_{k-j}. \tag{30}$$

Using $\bar{w}_{l-1}$ and $\bar{v}_{l-1}$ from (19) and Y and Z from (20) in n_k in (23)

$$n_k=\sum_{t=1}^{L}Y_tw_{t+k-L-1}+\sum_{t=1}^{L+1}Z_tv_{t+k-L-1}. \tag{31}$$

Using (30) and (31), $\mathbb{E}[x_kn_k^T]$ reads

$$\mathbb{E}[x_kn_k^T]=(A+BK)^k\underbrace{\mathbb{E}[x_0n_k^T]}_{=\mathbf{0}}$$
$$+\sum_{j=1}^{k-1}(A+BK)^{j-1}(BK\underbrace{\mathbb{E}[n_{k-j}n_k^T]}_{t_1}+G\underbrace{\mathbb{E}[w_{k-j}n_k^T]}_{t_2}).$$

Regarding the term t_1, one can see that based on (31), $\mathbb{E}[n_{k-j}n_k^T]=\mathbf{0}$ for $j>L$. For $j<L$, we have the following:

$$t_1=\mathbb{E}[n_{k-j}n_k^T]=\underbrace{\mathbb{E}\left[\sum_{t=1}^{L}Y_tw_{t+k-j-L-1}\sum_{t=1}^{L}w_{t+k-L-1}^TY_t^T\right]}_{t_{11}}$$
$$+\underbrace{\mathbb{E}\left[\sum_{t=1}^{L}Y_tw_{t+k-j-L-1}\sum_{t=1}^{L+1}v_{t+k-L-1}^TZ_t^T\right]}_{=\mathbf{0}}$$
$$+\underbrace{\mathbb{E}\left[\sum_{t=1}^{L+1}Z_tv_{t+k-j-L-1}\sum_{t=1}^{L}w_{t+k-L-1}^TY_t^T\right]}_{=\mathbf{0}}$$
$$+\underbrace{\mathbb{E}\left[\sum_{t=1}^{L+1}Z_tv_{t+k-j-L-1}\sum_{t=1}^{L+1}v_{t+k-L-1}^TZ_t^T\right]}_{t_{12}}.$$

Note that the second and third lines are zero because $\mathbb{E}[w_kv_j^T]=\mathbf{0}$. The terms t_{11} and t_{12} read

$$t_{11}=\mathbb{E}\left[\sum_{t=j+1}^{L}Y_tw_{t+k-j-L-1}\sum_{t=1}^{L-j}w_{t+k-L-1}^TY_t^T\right]$$
$$=\mathbb{E}\left[\sum_{t=1}^{L-j}Y_{t+j}w_{t+k-L-1}\sum_{t=1}^{L-j}w_{t+k-L-1}^TY_t^T\right]$$
$$=\sum_{t=1}^{L-j}Y_{t+j}W_wY_t^T$$
$$t_{12}=\mathbb{E}\left[\sum_{t=j+1}^{L+1}Z_tv_{t+k-j-L-1}\sum_{t=1}^{L-j+1}vw_{t+k-L-1}^TZ_t^T\right]$$
$$=\mathbb{E}\left[\sum_{t=1}^{L-j+1}Z_{t+j}v_{t+k-L-1}\sum_{t=1}^{L-j+1}v_{t+k-L-1}^TZ_t^T\right]$$
$$=\sum_{t=1}^{L-j+1}Z_{t+j}W_vZ_t^T.$$

For $j=L$, $t_{11}=\mathbf{0}$ and $t_{12}=Z_LW_vZ_1^T$. Regarding the term t_2, it is easy to verify that $t_2=\mathbf{0}$ for $j>L$. For $j\le L$

$$t_2=\sum_{t=1}^{L}\mathbb{E}[w_{k-j}w_{t+k-L-1}^T]Y_t^T+\sum_{t=1}^{L+1}\underbrace{\mathbb{E}[w_{k-j}v_{t+k-L-1}^T]}_{=\mathbf{0}}Z_t^T$$
$$=W_wY_{L-j+1}^T.$$

By inserting t_{11}, t_{12}, t_{13}, t_{14}, and t_2 into $\mathbb{E}[x_kn_k^T]$, we have

$$\mathbb{E}[x_kn_k^T]=\sum_{j=1}^{L-1}(A+BK)^{j-1}BK\sum_{t=1}^{L-j}Y_{t+j}W_wY_t^T$$
$$+\sum_{j=1}^{L-1}(A+BK)^{j-1}BK\sum_{t=1}^{L-j+1}Z_{t+j}W_vZ_t^T$$
$$+(A+BK)^{L-1}BKZ_LW_vZ_1^T$$
$$+\sum_{a=1}^{L}(A+BK)^{a-1}GW_wY_{L-a+1}^T$$

which is (26) after changing the index of the last summation.

References

[1] B. Recht, "A tour of reinforcement learning: The view from continuous control," *Annu. Rev. Control Robot. Auton. Syst.*, vol. 2, pp. 253–279, 2019.

[2] F. A. Yaghmaie and L. Ljung, "A crash course on reinforcement learning," 2021, *arXiv:2103.04910*.

[3] D. P. Bertsekas, *Reinforcement Learn. and Optimal Control*. Nashua, NH, USA: Athena Scientific, 2019.

[4] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, vol. 1, 2nd Ed., Cambridge, MA, USA: MIT Press, 2018. [Online]. Available: http://www.incompleteideas.net/book/the-book-2nd.html
[5] M. Fazel, R. Ge, S. M. Kakade, and M. Mesbahi, "Global convergence of policy gradient methods for the linear quadratic regulator," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1467–1476.
[6] F. A. Yaghmaie and H. Modares, "Online optimal tracking of linear systems with adversarial disturbances," *Trans. Mach. Learn. Res.*, 2022.
[7] L. Ljung, *System Identification - Theory the User*, 2nd Ed., Hoboken, NJ, USA: Prentice Hall, 1999.
[8] N. Niknejad, F. A. Yaghmaie, and H. Modares, "Online reference tracking for linear systems with unknown dynamics and unknown disturbances," *Trans. Mach. Learn. Res.*, 2023.
[9] K. J. Åström and B. Wittenmark, *Adaptive Control*, 2nd Ed., Hoboken, NJ, USA: Prentice Hall, 1994.
[10] F. A. Yaghmaie, F. Gustafsson, and L. Ljung, "Linear quadratic control using model-free reinforcement learning," *IEEE Trans. Autom. Control*, vol. 68, no. 2, pp. 737–752, Feb. 2023.
[11] F. A. Yaghmaie and F. Gustafsson, "Using reinforcement learning for model-free linear quadratic control with process and measurement noises," in *Proc. IEEE 58th Conf. Decis. Control*, 2019, pp. 6510–6517.
[12] M. Hausknecht and P. Stone, "Deep recurrent Q-learning for partially observable MDPs," in *Proc. AAAI Fall Symp. - Tech. Rep., vol. FS-15-06*, 2015, pp. 29–37.
[13] J. N. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," *Adv. Neural Inf. Process. Syst.*, pp. 2137–2145, 2016.
[14] P. Zhu, X. Li, P. Poupart, and G. Miao, "On improving deep reinforcement learning for POMDPs," 2018, *arXiv:1804.06309*.
[15] M. Simchowitz, "Making non-stochastic control (almost) as easy as stochastic," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, Art. no. 1538.
[16] S. Lale, K. Azizzadenesheli, B. Hassibi, and A. Anandkumar, "Logarithmic regret bound in partially observable linear dynamical systems," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 20876–20888.
[17] F. L. Lewis and K. G. Vamvoudakis, "Reinforcement learning for partially observable dynamic processes: Adaptive dynamic programming using measured output data," *IEEE Trans. Syst., Man, Cybern., Part B: Cybern.*, vol. 41, no. 1, pp. 14–25, Feb. 2011.
[18] F. L. Lewis, L. Xie, and D. Popa, *Optimal and Robust Estimation: With An Introduction to Stochastic Control Theory*. Boca Raton, FL, USA: CRC Press, 2017.
[19] T. Glad and L. Ljung, *Control Theory*. Boca Raton, FL, USA: CRC Press, 2018.
[20] D. Bertsekas, *Dynamic Programming and Optimal Control, Vol. I*: Bellmont, MA, USA: Athena Scientific, 2012.
[21] Y. Abbasi-Yadkori, N. Lazic, and C. Szepesvari, "Model-free linear quadratic control via reduction to expert prediction," in *Proc. 22nd Int. Conf. Artif. Intell. Statist.*, 2019, pp. 3108–3117.
[22] S. Tu and B. Recht, "Least-squares temporal difference learning for the linear quadratic regulator," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5005–5014.
[23] G. A. Hewer, "An iterative technique for the computation of the steady state gains for the discrete optimal regulator," *IEEE Trans. Autom. Control*, vol. AC-16, no. 4, pp. 382–384, Aug. 1971.
[24] B. Hambly, R. Xu, and H. Yang, "Policy gradient methods for the noisy linear quadratic regulator over a finite horizon," *SIAM J. Control Optim.*, vol. 59, no. 5, pp. 3359–3391, 2021.