Model-Free Reinforcement Learning of Minimal-Cost Variance Control

Gangshan Jing, He Bai, Jemin George, and Aranya Chakrabortty

***Abstract*—This letter proposes two reinforcement learning (RL) algorithms for solving a class of coupled algebraic Riccati equations (CARE) for linear stochastic dynamic systems with unknown state and input matrices. The CARE are formulated for a minimal-cost variance (MCV) control problem that aims to minimize the variance of a cost function while keeping its mean at an acceptable range using a noisy infinite-horizon full-state feedback linear quadratic regulator (LQR). We propose two RL algorithms where the input matrix can be estimated at the very first iteration. This, in turn, frees up significant amount of computational complexity in the intermediate steps of the learning phase by avoiding repeated matrix inversion of a high-dimensional data matrix. The overall complexity is shown to be less than RL for both stochastic and deterministic LQR. Additionally, the disturbance noise entering the model is not required to satisfy any condition for ensuring efficiency of either RL algorithms. Simulation examples are presented to illustrate the effectiveness of the two designs.**

***Index Terms*—Reinforcement learning, stochastic dynamic systems, variance control, coupled Riccati equations.**

I. Introduction

Reinforcement learning (RL) is a powerful learning method for seeking optimal policies of long-term indices by adjusting actions according to the reward from an environment [1]. In the context of control systems, this concept has been extensively used for learning linear quadratic regulator (LQR)-type optimal controls for linear time invariant (LTI) systems with unknown state and input matrices [2]–[5]. A specific method for RL, referred to as Approximate Dynamic Programming (ADP), has been used for solving model-free optimal control problems for continuous-time systems [6], discrete-time systems [7], and systems with disturbances [8].

In this letter, we focus on an important class of control problems in stochastic dynamical systems called minimum-cost variance (MCV) control. MCV deals with control scenarios when systems are influenced by stochastic disturbances, the purpose being to minimize a given cost function in terms of its variance. Examples of MCV include control of wind plants [9], satellite communication control [10], game-theoretic control [11], target-tracking of robotic swarms, etc. Early works on MCV control in continuous-time were reported in [12], which were later extended in [13] by using the Halmilton-Jacobi theory to derive a set of coupled algebraic Riccati equations (CARE) to solve for full-state feedback MCV. The connections between linear quadratic Gaussian (LQG) control, risk-sensitive control via cost cumulants, and MCV control were also shown. In [14] and [15], the authors proposed two iterative algorithms for solving the CARE, and two sufficient conditions for convergence of these algorithms, respectively. Sufficient conditions for existence and uniqueness of the solutions to the CARE were given in [14] and [16]. The work in [17] studied a more general problem of minimizing the linear combination of the first k cumulants of the cost function for LQR problems. However, all of these works are model-based, meaning that the control design is carried out assuming that the system dynamics are known exactly.

We develop model-free RL algorithms for infinite-horizon MCV control of linear stochastic systems with unknown system and input matrices. Our approach is to solve the CARE described in [13] using online measurements of the states, the inputs, and the process noise. In the literature, model-based optimal control for nonlinear systems with control-dependent noise was studied in [18]. Model-free LQR with control-dependent noise was investigated in [19]. However, to the best of our knowledge, no model-free RL algorithms have been proposed so far for MCV control. In the MCV control problem, the two coupled Riccati equations need to be solved, which intuitively requires more computational complexity for RL compared to conventional LQR. To reduce the computational complexity, we propose two algorithms where the input matrix is estimated at the first iteration of the RL. This is different from the RL algorithm in [19] which estimates the input matrix in every iteration. Because of this difference,

Manuscript received March 17, 2020; revised April 28, 2020; accepted May 13, 2020. Date of publication May 19, 2020; date of current version June 8, 2020. This work was supported in part by the CCDC Army Research Laboratory Director's Research Award for External Collaborative Initiative under Grant DEC19-SE-021. The work of Gangshan Jing and Aranya Chakrabortty was supported in part by NSF under Grant CNS 1931932. Recommended by Senior Editor G. Cherubini. *(Corresponding author: Gangshan Jing.)*

Gangshan Jing and Aranya Chakrabortty are with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27695 USA (e-mail: gjing@ncsu.edu; achakra2@ncsu.edu).

He Bai is with the Department of Mechanical and Aerospace Engineering, Oklahoma State University, Stillwater, OK 74078 USA (e-mail: he.bai@okstate.edu).

Jemin George is with the Computational and Information Sciences Directorate (CISD), CCDC Army Research Laboratory, Adelphi, MD 20783 USA (e-mail: jemin.george.civ@mail.mil).

Digital Object Identifier 10.1109/LCSYS.2020.2995547

our algorithms avoid repeated inversion of a high-dimensional data matrix, thereby leading to significant reduction in computational complexity in the intermediate steps of the learning phase. The overall complexity is shown to be less than that of RL for both stochastic LQR [19] and deterministic LQR [2]. Furthermore, our approach does not require the disturbance noise to satisfy any algebraic property such as those required in [19]. The MCV control is a general class of problem that covers both deterministic and stochastic LQR. Therefore, if needed, our proposed algorithms can be applied to those two problems as well with lower computational complexity.

The rest of this letter is organized as follows. Section II formulates MCV control; Section III proposes two RL algorithms for solving CARE; Section IV provides numerical simulations; Section V concludes this letter.

Notation: Throughout this letter, I_d is the $d \times d$ identity matrix; $\otimes$ means the Kronecker product; $\langle \cdot, \cdot \rangle$ is the matrix inner product defined as $\langle A, B\rangle = \mathrm{tr}(A^T B)$; $\mathrm{tr}(X)$ means the trace of matrix X; $X \succeq 0$ implies that X is positive semi-definite; $A \preceq B$ for two positive semi-definite matrices A and B implies that $B - A \succeq 0$; monotonic decrease in X_k implies that $X_{k+1} - X_k \succeq 0$ for $k = 1, 2, \ldots,$ $\mathrm{vec}(X) \in \mathbb{R}^{n^2}$ denotes the vector stacking columns of $X \in \mathbb{R}^{n\times n}$ in the order that they appear in X; $\mathrm{mat}(x)$ is a matrix such that $x = \mathrm{vec}(\mathrm{mat}(x))$; Given a symmetric matrix $X = [X_{ij}] \in \mathbb{R}^{n\times n}$, we define the mapping $\nu(\cdot) : \mathbb{R}^{n\times n} \to \mathbb{R}^{\frac{n(n+1)}{2}}$ as $\nu(X) = (X_{11}, 2X_{12}, \ldots, 2X_{1n}, X_{22}, 2X_{23}, \ldots, 2X_{2n}, \ldots, X_{nn})^\top$. It is easy to see that there exists a constant matrix $\Phi \in \mathbb{R}^{n^2\times \frac{n(n+1)}{2}}$ such that $\mathrm{vec}(X) = \Phi\nu(X)$ for any $X \in \mathbb{R}^{n\times n}$.

II. Problem Statement

A. MCV Problem Formulation

Consider the following linear stochastic differential equation (SDE):

$$dx = (Ax + Bu)dt + Gdw, \quad x(0) = x_0, \tag{1}$$

where $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$ are the state and the control input, respectively. System matrix $A \in \mathbb{R}^{n\times n}$ and input matrix $B \in \mathbb{R}^{n\times m}$ are unknown, but their dimensions are known. The system is excited by stochastic noise induced by a stationary Wiener process $w \in \mathbb{R}^q$ with zero mean and the correlation of increments

$$E\Big[(w(t_1) - w(t_2))(w(t_1) - w(t_2))^\top\Big] = W|t_1 - t_2|,$$

where, $E[\,\cdot\,]$ denotes the expectation function, and W is a positive definite matrix. Suppose $G \in \mathbb{R}^{n\times q}$ is known, and the noise Gdw and the state x are both measurable.[1] The MCV control objective is to design u such that the weighted sum of the mean and the variance (the 2nd cumulant) of the following objective function is minimized:

$$J(x, u, t_f) = \int_0^{t_f} (x^\top Qx + u^\top Ru)d\tau. \tag{2}$$

[1]The noise has to be measurable so that the input and state data can describe the system accurately. Similar assumption has been made in [19].

More specifically, the objective of an infinite-horizon MCV control is to minimize

$$\mathcal{J}(x, u) = \lim_{t_f\to\infty} \frac{E\big[J(x, u, t_f)\big]}{t_f} + \gamma \lim_{t_f\to\infty} \frac{\mathrm{Var}\big[J(x, u, t_f)\big]}{t_f},$$

where γ is a positive constant, and $\mathrm{Var}[\,\cdot\,]$ denotes variance.

Lemma 1 [14]: The optimal control law for the infinite-horizon optimal control problem has the form

$$u = Kx = -R^{-1}B^\top(M + \gamma H)x, \tag{3}$$

where $\gamma > 0$, and M and H satisfy the following CARE:

$$A^\top M + MA + Q - MBR^{-1}B^\top M + \gamma^2 HBR^{-1}B^\top H = 0, \tag{4}$$
$$\begin{aligned} &A^\top H + HA - MBR^{-1}B^\top H - HBR^{-1}B^\top M \\ &\quad - 2\gamma HBR^{-1}B^\top H + 4MGWG^\top M = 0. \end{aligned} \tag{5}$$

Lemma 1 can be viewed as a special case of [17, Th. 4.4.1] with $k = 2$. The parameter $\gamma > 0$ can be viewed as the weight of the second cumulant (variance) in the objective function, which implies that the larger γ is, the smaller the variance corresponding to the optimal solution is. If there is a unique pair (M, H) satisfying the above CARE, then the corresponding controller will be optimal. Otherwise any pair (M, H) satisfying (4-5) is a local optimal controller [14], [17]. Conditions for uniqueness of the solution to (4-5) were presented in [14], [16]. For our design we will solve the CARE (4)-(5) when A and B are unknown.

Remark 1: Note that the MCV control problem reduces to the deterministic LQR problem when G is a zero matrix, and to a mean cost minimization problem when $\gamma = 0$. As a result, the algorithms we propose in this letter can alternatively be applied to these two problems as well.

B. Two Model-Based Iterative Algorithms

Substituting K from (3) into (4)-(5) yields

$$(A + BK)^\top M + M(A + BK) + K^\top RK + Q = 0. \tag{6}$$
$$(A + BK)^\top H + H(A + BK) + 4MGWG^\top M = 0. \tag{7}$$

Similar to the Kleinman's algorithm [20] for solving the conventional algebraic Riccati equations that arise in LQR, two iterative algorithms for solving the CARE in (4)-(5) are presented in [14] and [15]. For easy referencing, we recall these two algorithms in Algorithm 1 and Algorithm 2, respectively. However, compared to the Kleinman's algorithm, even when K_0 is chosen to be stabilizing, M_k and H_k obtained from Algorithms 1 and 2 may not be monotonically decreasing, and so K_k may become destabilizing at some step $k \geq 1$. In [14] and [15], the authors proposed conditions for convergence under which the matrix $M_k + \gamma H_k$ is monotonically decreasing, and K_k is guaranteed to be stabilizing. Accordingly, we will develop two RL algorithms that will recover the solutions of Algorithm 1 and Algorithm 2, while matrices A and B are unknown. Note that both Algorithms 1 and 2 can be viewed as *policy iteration* algorithms [6], where step 2 and step 3 correspond to policy evaluation, while step 4 corresponds to policy improvement.

A common property of Algorithm 1, Algorithm 2 and Kleinman's algorithm [20] is that they all require the initial

Algorithm 1 Iterative Algorithm for MCV From [14]

```
1. Choose an initial gain $K_0$ such that $A_0 = A + BK_0$ is stable. Let $k = 0$.
2. Obtain $M_k$ by solving
       $A_k^\top M_k + M_k A_k + K_k^\top R K_k + Q = 0,$
   where $A_k = A + BK_k$.
3. Obtain $H_k$ by solving
       $A_k^\top H_k + H A_k + 4 M_k G W G^\top M_k = 0.$
4. Compute $K_{k+1} = -R^{-1} B^\top (M_k + \gamma H_k)$.
5. For a given threshold $\epsilon > 0$, check the condition:
       $\sigma \triangleq \frac{\|K_{k+1} - K_k\|}{\|K_k\|} < \epsilon.$
6. If the above condition does not hold, then $k \leftarrow k + 1$ and go back to step 2; Otherwise the optimal controller is $u^* = K_{k+1} x$.
```

Algorithm 2 Iterative Algorithm for MCV From [15]

```
1. Choose an initial gain $K_0$ such that $A_0 = A + BK_0$ is stable, and choose a positive semi-definite matrix $M_0$. Let $k = 1$.
2. Obtain $M_k$ by solving
       $A_{k-1}^\top M_k + M_k A_{k-1} + K_{k-1}^\top R K_{k-1} + Q = 0,$
   where $A_{k-1} = A + BK_{k-1}$.
3. Obtain $H_k$ by solving
       $A_{k-1}^\top H_k + H A_{k-1} + 4 M_{k-1} G W G^\top M_{k-1} = 0.$
4. Compute $K_k = -R^{-1} B^\top (M_k + \gamma H_k)$.
5. For a given threshold $\epsilon > 0$, check the condition:
       $\sigma \triangleq \frac{\|K_k - K_{k-1}\|}{\|K_k\|} < \epsilon.$
6. If the above condition does not hold, then $k \leftarrow k + 1$ and go back to step 2; Otherwise the optimal controller is $u^* = K_k x$.
```

control gain K_0 to be stabilizing. These algorithms will either not converge or converge to an incorrect matrix if K_0 is destabilizing. Accordingly, the RL algorithms that we will propose based on Algorithms 1 and 2 will require K_0 to be stabilizing as well, similar to the assumption made for the RL algorithms reported in [2], [3], [5], [19].

Remark 2: The main differences between Algorithm 1 and Algorithm 2 are as follows: 1) the first step of Algorithm 2 requires an artificially selected M_0, which may influence its convergence; Algorithm 1, however, does not need a matrix M_0 at the first step; 2) the update steps for H_k are different for the two algorithms. At step k in any iteration of Algorithm 1 the update of H_k requires one to solve M_k in advance, while in Algorithm 2 the update of H_k is based on M_{k-1}, and so H_k and M_k can be updated simultaneously. As a result, one advantage of Algorithm 2 is that the two Lyapunov equations can be solved in parallel. Moreover, Algorithm 1 and Algorithm 2 may be efficient for different classes of MCV problems [14].

III. RL Algorithms for MCV Control

In this section, we present two off-policy algorithms that can recover the solutions of Algorithm 1 and Algorithm 2 when matrices A and B are unknown.

A. RL Algorithm for Iterative Algorithm 1

To design the RL algorithm, we will transform the model-based Lyapunov equations in Algorithm 1 to a set of equations that are based on input and state data only by utilizing the system dynamics (1). We first differentiate $x^\top M_k x$ and $x^\top H_k x$. By Ito's lemma (see 5.3.9b in [22]) and using the Lyapunov equations in Algorithm 1, we get

$$\begin{aligned} d(x^\top M_k x) &= x^\top (A_k^\top M_k + M_k A_k) x dt + 2x^\top M_k B u dt \\ &\quad - 2x^\top M_k B K_k x dt + 2x^\top M_k G dw + \operatorname{tr}(G^\top M_k G) dt \\ &= -x^\top K_k^\top R K_k x dt - x^\top Q x dt + 2x^\top M_k B u dt \\ &\quad - 2x^\top M_k B K_k x dt + 2x^\top M_k G dw + \sum_{i=1}^{n}\sum_{j=1}^{n} g_i^\top M_k g_j dt, \end{aligned} \tag{8}$$

and

$$\begin{aligned} d(x^\top H_k x) &= x^\top (A_k^\top H_k + H_k A_k) x dt + 2x^\top H_k B u dt \\ &\quad - 2x^\top H_k B K_k x dt + 2x^\top H_k G dw + \operatorname{tr}(G^\top H_k G) dt \\ &= -4x^\top M_k G W G^\top M_k x dt + 2x^\top H_k B u dt \\ &\quad - 2x^\top H_k B K_k x dt + 2x^\top H_k G dw + \sum_{i=1}^{n}\sum_{j=1}^{n} g_i^\top H_k g_j dt, \end{aligned} \tag{9}$$

where $g_1, \ldots, g_n$ are column vectors forming G. Integrating both sides of (8) on time interval $[t_1, t_2]$, we get

$$\begin{aligned} \int_{t_1}^{t_2} d(x^\top M_k x) &= -\int_{t_1}^{t_2} x^\top K_k^\top R K_k x dt - \int_{t_1}^{t_2} x^\top Q x dt \\ &\quad + \int_{t_1}^{t_2} 2x^\top M_k B u dt - \int_{t_1}^{t_2} 2x^\top M_k B K_k x dt \\ &\quad + \int_{t_1}^{t_2} 2x^\top M_k G dw + \int_{t_1}^{t_2} \sum_{i=1}^{n}\sum_{j=1}^{n} g_i^\top M_k g_j dt, \end{aligned}$$

which is equivalent to

$$\begin{aligned} \langle \int_{t_1}^{t_2} d(xx^\top), M_k \rangle &= \langle \int_{t_1}^{t_2} xx^\top dt, K_k^\top R K_k - Q \rangle \\ &\quad + \langle \int_{t_1}^{t_2} 2ux^\top dt, M_k B \rangle - \langle \int_{t_1}^{t_2} 2xx^\top dt, M_k B K_k \rangle \\ &\quad + \langle \int_{t_1}^{t_2} 2G dw x^\top, M_k \rangle + \langle \int_{t_1}^{t_2} \sum_{i=1}^{n}\sum_{j=1}^{n} g_j g_i^\top dt, M_k \rangle. \end{aligned}$$

Similarly, for (9), we have

$$\begin{aligned} \langle \int_{t_1}^{t_2} d(xx^\top), H_k \rangle &= -\langle \int_{t_1}^{t_2} 4xx^\top dt, M_k G W G^\top M_k \rangle \\ &\quad + \langle \int_{t_1}^{t_2} 2ux^\top dt, H_k B \rangle - \langle \int_{t_1}^{t_2} 2xx^\top dt, H_k B K_k \rangle \\ &\quad + \langle \int_{t_1}^{t_2} 2G dw x^\top, H_k \rangle + \langle \int_{t_1}^{t_2} \sum_{i=1}^{n}\sum_{j=1}^{n} g_j g_i^\top dt, H_k \rangle. \end{aligned}$$

We further define

$$\delta_{xx} = \left[x(t)\otimes x(t)|_{t_0}^{t_1}, \ldots, x(t)\otimes x(t)|_{t_{p-1}}^{t_p}\right]^\top \in \mathbb{R}^{p\times n^2},$$

$$I_{xx} = \left[\int_{t_0}^{t_1} x\otimes x d\tau, \ldots, \int_{t_{p-1}}^{t_p} x\otimes x d\tau\right]^\top \in \mathbb{R}^{p\times n^2},$$

$$I_{xu} = \left[\int_{t_0}^{t_1} x\otimes u d\tau, \ldots, \int_{t_{p-1}}^{t_p} x\otimes u d\tau\right]^\top \in \mathbb{R}^{p\times mn},$$

$$I_{xw} = \left[\int_{t_0}^{t_1} x\otimes G dw, \ldots, \int_{t_{p-1}}^{t_p} x\otimes G dw\right]^\top \in \mathbb{R}^{p\times n^2},$$

$$\delta_{gg} = \sum_{i=1}^{n}\sum_{j=1}^{n}\left[\int_{t_0}^{t_1} g_i\otimes g_j dt, \ldots, \int_{t_{p-1}}^{t_p} g_i\otimes g_j dt\right]^\top \in \mathbb{R}^{p\times n^2}.$$

As Gdw is known, combining (8) and (9), we have

$$\Theta_k\begin{pmatrix}\text{vec}(M_k)\\ \text{vec}(B^\top M_k)\end{pmatrix} = \Xi_k, \tag{10}$$

where

$$\Theta_k = \left[\delta_{xx} - 2I_{xw} - \delta_{gg}, -2I_{xu} + 2I_{xx}(I_n\otimes K_k^\top)\right] \in \mathbb{R}^{p\times(n^2+mn)},$$
$$\Xi_k = -I_{xx}\text{vec}(K_k^\top R K_k + Q) \in \mathbb{R}^p.$$

Moreover,

$$\Theta_k\begin{pmatrix}\text{vec}(H_k)\\ \text{vec}(B^\top H_k)\end{pmatrix} = \Omega_k, \tag{11}$$

where $\Omega_k = -4I_{xx}\text{vec}(M_k G W G^\top M_k) \in \mathbb{R}^p$.

Following similar approach as in [19], an iterative RL algorithm can be easily derived. However, we notice that solving (10) and (11) in the least squares sense would be time-consuming, especially if the system order n is large. We next show that the dimension of the least squares equations to be solved in the RL algorithm can be reduced extensively by estimating B in the first iteration.

We define I_x such that

$$I_{xx}\text{vec}(X) = I_x\nu(X)$$

where $I_x = I_{xx}\Phi = [\int_{t_0}^{t_1}\mu(x)d\tau, \ldots, \int_{t_{p-1}}^{t_p}\mu(x)dt]^\top$. Here the mapping $\mu(\cdot): \mathbb{R}^n \to \mathbb{R}^{\frac{n(n+1)}{2}}$ on vector $y = (y_1, \ldots, y_n)^\top \in \mathbb{R}^n$ is defined as:

$$\mu(y) = (y_1^2, y_1y_2, \ldots, y_1y_n, y_2^2, y_2y_3, \ldots, y_{n-1}y_n, y_n^2)^\top.$$

Note that matrix B can be estimated by solving the least squares problem only once. Once B is known, the variable $B^\top M_k$ can be eliminated as shown in the following lemma.

Lemma 2: Suppose $\text{rank}[I_x, I_{xu}] = \frac{n(n+1)}{2} + mn$. Let $\begin{pmatrix}\text{vec}(M)\\ \text{vec}(N)\end{pmatrix}$ be the solution of (10) with $k = 0$, then $B = M^{-1}N^\top$.

Proof: From the definitions of Θ_0 and Ξ_0, (10) implies that

$$(\delta_{xx} - 2I_{xw} - \delta_{gg})\text{vec}(M) + [-2I_{xu} + 2I_{xx}(I_n\otimes K_0^\top)]\text{vec}(N)$$
$$= -I_{xx}\text{vec}(K_0^\top R K_0 + Q. \tag{12}$$

Algorithm 3 RL Algorithm Corresponding to Algorithm 1

```
1. Given an initial stabilizing gain $K_0$. Let $k = 0$.
2. Apply $u = K_0x + e(t)$ to (1) on time interval $[t_0, t_p]$ until
   rank $[I_{xx}, I_{xu}] = \frac{n(n+1)}{2} + mn$. Compute $\Theta_0$ and $\Xi_0$. Obtain
   the solution $\begin{pmatrix}\text{vec}(M)\\ \text{vec}(N)\end{pmatrix}$ to (10) with $k = 0$. Compute
   $B = M^{-1}N^\top$, and let $M_k = M$.
3. Compute $\Lambda_k$, $\Xi_k$, $\Omega_k$ and $T_k = (\Lambda_k^\top\Lambda_k)^{-1}\Lambda_k^\top$.
4. Compute $M_k = \text{mat}(T_k\Xi_k)$, $H_k = \text{mat}(T_k\Omega_k)$, and
   $K_{k+1} = -R^{-1}B^\top(M_k + \gamma H_k)$ successively. For given
   threshold $\epsilon > 0$, check the following condition
       $\sigma \triangleq \frac{\|K_{k+1} - K_k\|}{\|K_k\|} < \epsilon.$
   If the condition does not hold, let $k \leftarrow k+1$ and go back
   to Step 3; Otherwise go to Step 5.
5. The optimal controller is $u^* = K_{k+1}x$.
```

By using the first equality in (8), we have

$$\delta_{xx}\text{vec}(M) = I_{xx}\text{vec}(A_0^\top M + MA_0) + 2I_{xu}\text{vec}(B^\top M)$$
$$- 2I_{xx}\text{vec}(K_0^\top B^\top M) + 2I_{xw}\text{vec}(M) + \delta_{gg}\text{vec}(M). \tag{13}$$

Combining (12) and (13), yields

$$I_{xx}\text{vec}(A_0^\top M + MA_0 + K_0^\top RK_0 + Q + 2K_0^\top N - 2K_0^\top B^\top M)$$
$$+ 2I_{xu}\text{vec}(N - B^\top M) = 0,$$

which is equivalent to

$$I_x\nu(A_0^\top M + MA_0 + K_0^\top RK_0 + Q + 2K_0^\top N - 2K_0^\top B^\top M)$$
$$+ 2I_{xu}\text{vec}(N - B^\top M) = 0.$$

Since $[I_x, I_{xu}]$ is of full column rank, we have $N = B^\top M$. ■

By virtue of Lemma 2, we can estimate B after the first iteration $k = 0$. For $k \geq 1$, equations (10) and (11) can be modified as the following two equations:

$$\Lambda_k\text{vec}(M_k) = \Xi_k \tag{14}$$
$$\Lambda_k\text{vec}(H_k) = \Omega_k \tag{15}$$

where $\Lambda_k \in \mathbb{R}^{p\times n^2}$ is given by

$$\Lambda_k = \delta_{xx} - 2I_{xw} - \delta_{gg} - 2I_{xu}(I_n\otimes B^\top) + 2I_{xx}(I_n\otimes K_k^\top B^\top).$$

In this case, the most time-consuming part of the least squares step is computing the inverse of $\Lambda_k^\top\Lambda_k \in \mathbb{R}^{n^2\times n^2}$. In the original least squares problem, one has to compute the inverse of $\Theta_k^\top\Theta_k \in \mathbb{R}^{(n^2+mn)\times(n^2+mn)}$. Since the computational complexity associated with inverting an n-dimensional matrix is $\mathcal{O}(n^{2.376})$ [21], by reducing the dimension of the least squares problems, the computational complexity is significantly reduced.

Let $e(t)$ be the exploration noise applied in the control input at time t, which is used to generate input and state information such that the rank condition in Lemma 2 can be satisfied. The RL algorithm is given in Algorithm 3. Note that Algorithm 3 is an off-policy algorithm since the control input applied to the system for generating data is independent of the policy updated during each iteration.

Remark 3: The main innovation of our proposed RL is that although there are two Riccati equations to solve in Algorithm 1, estimating B in the very first step in Algorithm 3 allows us to solve the inverse of $\Lambda_k^\top \Lambda_k \in \mathbb{R}^{n^2\times n^2}$ only once per iteration, otherwise we will have to solve the inverse of $\Theta_k^\top \Theta_k \in \mathbb{R}^{(n^2+mn)\times(n^2+mn)}$ during each iteration. Thus, estimating B saves significant amount of computational complexity. For example, the dimension of the same matrix that needs to be inverted in [2] is $\frac{1}{2}n(n+1)+mn$, and that in [19] is $2n^2+mn$. Our algorithm, therefore, will be significantly faster than RL for noisy LQR [19], and usually faster than RL for deterministic LQR [2]. This also implies that Algorithm 3 is advantageous when it is applied to deterministic LQR or mean minimization for noisy LQR. The same is true for Algorithm 4 that will be presented next.

Theorem 1: If rank $[I_x, I_{xu}] = \frac{n(n+1)}{2} + mn$ for each $k \in \mathbb{N}$, then rank$(\Lambda_k) = n^2$ for each $k \in \mathbb{N}$. Moreover, with the same initial condition, Algorithm 1 and Algorithm 3 generate same M_k and H_k at each step k.

The proof for Theorem 1 is similar to that in [2], and therefore is omitted here.

It is important to note that unlike [19], which also addresses a model-free noisy LQR problem, the rank condition here is independent of the noise induced term I_{xw}. To satisfy the rank condition in Algorithm 3, one should choose an exploration noise $e(t)$ to be persistently exciting, and collect data for long enough time. Usually $e(t)$ is chosen as a sum of sinusoidal functions with a sufficient number of distinct frequencies.

B. RL Algorithm for Iterative Algorithm 2

Similar to the first design, for Algorithm 2 we have

$$d(x^\top M_k x) = -x^\top K_{k-1}^\top R K_{k-1} x dt - x^\top Q x dt + 2x^\top M_k B u$$
$$dt - 2x^\top M_k B K_{k-1} x dt + 2x^\top M_k G dw + \sum_{i=1}^{n}\sum_{j=1}^{n} g_i^\top M_k g_j dt$$

and

$$d(x^\top H_k x) = -4x^\top M_{k-1} G W G^\top M_{k-1} x dt + 2x^\top H_k B u dt$$
$$-2x^\top H_k B K_{k-1} x dt + 2x^\top H_k G dw + \sum_{i=1}^{n}\sum_{j=1}^{n} g_i^\top H_k g_j dt.$$

By defining δ_{xx}, I_{xx}, I_{xw}, I_{xu} and δ_{gg} the same as before, we get

$$\Theta_k \begin{pmatrix} \mathrm{vec}(M_k) & \mathrm{vec}(H_k) \\ \mathrm{vec}(B^\top M_k) & \mathrm{vec}(B^\top H_k) \end{pmatrix} = (\Xi_k, \Omega_k) \tag{16}$$

where $\Theta_k = [\delta_{xx} - 2I_{xw} - \delta_{gg}, -2I_{xu} + 2I_{xx}(I_n \otimes K_{k-1}^\top)] \in \mathbb{R}^{p\times(n^2+mn)}$, $\Xi_k = -I_{xx}\mathrm{vec}(K_{k-1}^\top R K_{k-1} + Q) \in \mathbb{R}^p$, $\Omega_k = -4I_{xx}\mathrm{vec}(M_{k-1} G W G^\top M_{k-1}) \in \mathbb{R}^p$. Note that different from Algorithm 1, where matrices M_k and H_k have to be obtained sequentially, in Algorithm 2, M_k and H_k can be computed simultaneously. This is because in Algorithm 2, Ω_k is based on M_{k-1} instead of M_k.

Algorithm 4 RL Algorithm Corresponding to Algorithm 2

```
1. Start with an initial stabilizing gain $K_0$. Let $k = 1$.
2. Apply $u = K_0 x + e(t)$ to (1) on time interval $[t_0, t_p]$ until
   rank $[I_{xx}, I_{xu}] = \frac{n(n+1)}{2} + mn$. Choose a positive semi-definite $M_0$,
   compute $\Theta_1$, $\Xi_1$ and $\Omega_1$. Obtain the solution
   $\begin{pmatrix} \mathrm{vec}(M) & \mathrm{vec}(U) \\ \mathrm{vec}(N) & \mathrm{vec}(V) \end{pmatrix}$ to (16) with $k = 1$.
   Compute $B = M^{-1}N^\top$, $K_1 = -R^{-1}B^\top(M + \gamma U)$ and let $k \leftarrow k+1$.
3. Compute $\Lambda_k$, $\Xi_k$ and $\Omega_k$, obtain $M_k$ and $H_k$ by solving
   (17) in the least square sense.
4. Compute $K_k = -R^{-1}B^\top(M_k + \gamma H_k)$. For given threshold
   $\epsilon > 0$, check the following condition
       $\sigma \triangleq \frac{\|K_k - K_{k-1}\|}{\|K_k\|} < \epsilon.$
   If the condition does not hold, let $k \leftarrow k+1$ and go back
   to Step 3; Otherwise, go to Step 5.
5. The optimal controller is $u^* = K_k x$.
```

Similar to Lemma 2, matrix B can still be estimated in the first iteration. The difference is that unlike Algorithm 1, B in this case will be estimated from (16) with $k = 1$ (instead of $k = 0$). Once B is obtained, (16) is reduced to

$$\Lambda_k \big(\mathrm{vec}(M_k) \quad \mathrm{vec}(H_k)\big) = (\Xi_k, \Omega_k) \tag{17}$$

where $\Lambda_k = \delta_{xx} - 2I_{xw} - \delta_{gg} - 2I_{xu}(I_n \otimes B^\top) + 2I_{xx}(I_n \otimes K_{k-1}^\top B^\top) \in \mathbb{R}^{p\times n^2}$. The RL for Algorithm 2 is given in Algorithm 4.

Theorem 2: If rank $[I_x, I_{xu}] = \frac{n(n+1)}{2} + mn$ for each $k \in \mathbb{N}$, then rank$(\Lambda_k) = n^2$ for each $k \in \mathbb{N}$. Moreover, with the same initial configuration, Algorithm 2 and Algorithm 4 generate same M_k and H_k at each step k.

IV. Simulation Results

We consider a 2^{nd}-order example. Although the system size is small, we still choose this example as it has been used as a benchmark example for MCV control in [14], [15]. Consider $A = \begin{pmatrix} 1 & \frac{1}{8} \\ 0 & 4 \end{pmatrix}$, $B = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}$, $G = \frac{1}{3}I_2$, $Q = 4I_2$, $R = W = I_2$.

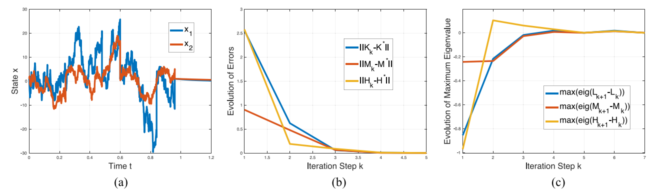
We set $\gamma = 3/8$, the initial gain $K_0 = \mathrm{diag}\{-2, 3\}$, and the excitation noise $e(t) = 100\sum_{i=1}^{100} \sin(w_i t)$, where w_i is selected from $[-500, 500]$ randomly, $i = 1, \ldots, 100$.

To compare our RL algorithms with their model-based counterparts we first run Algorithm 1 with $\epsilon = 10^{-5}$. It is observed that M and K converge to

$$M^* = \begin{pmatrix} 3.2594 & 0.0519 \\ 0.0519 & 2.4175 \end{pmatrix}, \quad K^* = \begin{pmatrix} -3.5995 & -0.0581 \\ -0.1161 & -4.9976 \end{pmatrix}.$$

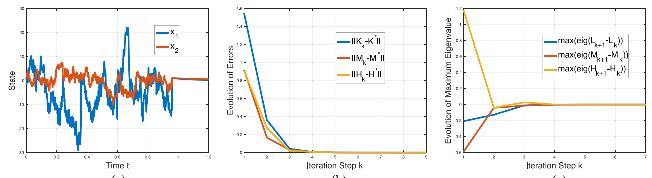
Next, we run Algorithm 3. The closed-loop response of $x(t)$ is shown in Fig. 1(a). It can be seen that K_k converges to a stabilizing gain. Fig. 1(b) shows that the algorithm converges to the same result as the model-based Algorithm 1. Fig. 1(c) shows that during the learning phase both M_k and $L_k = M_k + \gamma H_k$ are monotonically decreasing, but H_k is not monotonic.

To run Algorithm 4, we set $M_0 = \mathbf{0}_{2\times 2}$. We find that Algorithm 2 and Algorithm 4 converge to the same M^* and



(a) (b) (c)

Fig. 1. (a). Evolution of x under Algorithm 3; (b). Evolution of $\|K_k - K^*\|$, $\|M_k - M^*\|$ and $\|H_k - H^*\|$ under Algorithm 3; (c). Evolution of maximum eigenvalues of $M_{k+1} - M_k$ and $L_{k+1} - L_k$ under Algorithm 3.



(a) (b) (c)

Fig. 2. (a). Evolution of x under Algorithm 4; (b). Evolution of $\|K_k - K^*\|$, $\|M_k - M^*\|$ and $\|H_k - H^*\|$ under Algorithm 4; (c). Evolution of maximum eigenvalues of $L_{k+1} - L_k$, $M_{k+1} - M_k$ and $H_{k+1} - H_k$ under Algorithm 4.

K^* as those obtained by Algorithm 1. We plot the closed-loop trajectories of the state vector $x(t)$ in Fig. 2(a), evolution of the errors $\|K_k - K^*\|$, $\|M_k - M^*\|$ and $\|H_k - H^*\|$ in Fig. 2(b), and evolution of the maximum eigenvalues of $L_{k+1} - L_k$, $M_{k+1} - M_k$ and $H_{k+1} - H_k$ in Fig. 2(c). It can be observed that, although starting from a different initial point, and using a different updating mechanism, Algorithm 2 and Algorithm 4 eventually obtain the same optimal control gain as that obtained by Algorithm 1 and Algorithm 3.

We also tested a 10-dimensional example to examine the computational efficiency of the proposed algorithms. We skip the details of that example here due to space constraints. For that example, Algorithms 3 and 4 take 0.1000s and 0.1239s to complete, respectively; if we do not estimate B in the first step and replace matrix Λ_k in Step 4 of Algorithms 3 and 4 by Θ_k, the two algorithms take 0.2141s and 0.2175s, respectively. This reconfirms how estimating B in the first iteration can reduce the computational complexity.

V. Conclusion

We developed a set of off-policy RL algorithms for solving CARE encountered in the model-free MCV control problems. The specificity of the CARE is exploited to make the designs numerically cheap. Simulations are provided to demonstrate the numerical effectiveness as well as the ability of the RL algorithms to recover the optimal control gains produced by their model-based counterparts.

References

[1] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[2] Y. Jiang and Z. P. Jiang, "Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics," *Automatica*, vol. 48, no. 10, pp. 2699–2704, 2012.

[3] S. Mukherjee, H. Bai, and A. Chakrabortty, "On model-free reinforcement learning of reduced-order optimal control for singularly perturbed systems," in *Proc. IEEE Conf. Decis. Control*, 2018, pp. 5288–5293.

[4] H. Bai, J. George, and A. Chakrabortty, "Hierarchical control of multi-agent systems using online reinforcement learning," in *Proc. Amer. Control Conf. (ACC)*, Denver, CO, USA, Jul. 2020, pp. 246–253.

[5] D. Vrabie, O. Pastravanu, M. AbuKhalaf, and F. L. Lewis, "Adaptive optimal control for continuous-time linear systems based on policy iteration," *Automatica*, vol. 45, no. 2, pp. 477–484, 2009.

[6] Y. Yang, D. Wunsch, and Y. Yin, "Hamiltonian-driven adaptive dynamic programming for continuous nonlinear dynamical systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 8, pp. 1929–1940, Aug. 2017.

[7] Y. Yang, Z. Guo, H. Xiong, D. Ding, Y. Yin, and D. C. Wunsch, "Data-driven robust control of discrete-time uncertain linear systems via off-policy reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 12, pp. 3735–3747, Dec. 2019.

[8] Y. Yang, K. G. Vamvoudakis, H. Ferraz, and H. Modares, "Dynamic intermittent Q-learning-based model-free suboptimal co-design of L_2-stabilization," *Int. J. Robust Nonlinear Control*, vol. 29, no. 9, pp. 2673–2694, 2019.

[9] I. Filip, C. Vasar, I. Szeidert, and O. Prostean, "Self-tuning strategy for a minimum variance control system of a highly disturbed process," *Eur. J. Control*, vol. 46, pp. 49–62, Mar. 2019.

[10] K. D. Pham, "Assured satellite communications: A minimal-cost-variance system controller paradigm," in *Proc. Amer. Control Conf.*, 2016, pp. 6555–6561.

[11] F. Saleheen and C. Won, "Statistical Stackelberg game control: Open-loop minimal cost variance case," *Automatica*, vol. 101, pp. 338–344, Mar. 2019.

[12] M. K. Sain, "On minimal-variance control of linear systems with quadratic loss," Ph.D dissertation, Dept. Comput. Sci., Univ. Illinois at Urbana–Champaign, Champaign, IL, USA, 1965.

[13] M. K. Sain, C. H. Won, and B. F. Spencer, "Cumulants in risk-sensitive control: The full-state-feedback cost variance case," in *Proc. IEEE Conf. Decis. Control*, 1995, pp. 1036–1041.

[14] C. H. Won, M. K. Sain, and S. R. Liberty, "Infinite-time minimal cost variance control and coupled algebraic Riccati equations," in *Proc. Amer. Control Conf.*, 2003, pp. 5155–5160.

[15] G. Freiling, S.-R. Lee, and G. Jank, "Coupled matrix Riccati equations in minimal cost variance control problems," *IEEE Trans. Autom. Control*, vol. 44, no. 3, pp. 556–560, Mar. 1999.

[16] L. Cherfi, "New results in the Lyapunov-type algorithm for algebraic MCV Riccati equations," *Math. Comput. Model.*, vol. 51, nos. 3–4, pp. 150–155, 2010.

[17] K. D. Pham, "Statistical control paradigms for structural vibration suppression," Ph.D. dissertation, Dept. Elect. Eng., Univ. Notre Dame, Notre Dame, IN, USA, 2004.

[18] K. S. Bakshi, D. D. Fan, and E. A. Theodorou, "Stochastic control of systems with control multiplicative noise using second order FBSDEs," in *Proc. Amer. Control Conf. (ACC)*, 2017, pp. 424–431.

[19] T. Bian, Y. Jiang, and Z.-P. Jiang, "Adaptive dynamic programming for stochastic systems with state and control dependent noise," *IEEE Trans. Autom. Control*, vol. 61, no. 12, pp. 4170–4175, Dec. 2016.

[20] D. Kleinman, "On an iterative technique for Riccati equation computations," *IEEE Trans. Autom. Control*, vol. AC-13, no. 1, pp. 114–115, Feb. 1968.

[21] T. H. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, MA, USA: MIT Press, 2009.

[22] L. Arnold, *Stochastic Differential Equations: Theory and Applications*. New York, NY, USA: Wiley, 1974.