Computers and Chemical Engineering 177 (2023) 108325

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/cace

Unconstrained feedback controller design using Q-learning from noisy process data

Pratyush Kumar *, James B. Rawlings

Department of Chemical Engineering, University of California, Santa Barbara, CA 93106, United States

ARTICLE INFO

Keywords:
Reinforcement learning
Q-learning
Least squares policy iteration
System identification
Maximum likelihood estimation
Linear quadratic regulator

ABSTRACT

This paper develops a novel model-free Q-learning based approach to estimate linear, unconstrained feedback controllers from noisy process data. The proposed method is based on an extension of an available approach developed to estimate the linear quadratic regulator (LQR) for linear systems with full state measurements driven by Gaussian process noise of known covariance. First, we modify the approach to treat the case of an unknown noise covariance. Then, we use the modified approach to estimate a feedback controller for linear systems with both process and measurement noise and only output measurements. We also present a model-based maximum likelihood estimation (MLE) approach to determine a linear dynamic model and noise covariances from data, which is used to construct a regulator and state estimator for comparisons in simulation studies. The performances of the model-free and model-based controller estimation approaches are compared with an example heating, ventilation, and air-conditioning (HVAC) system. We show that the proposed Q-learning approach estimates a reasonably accurate feedback controller from 24 h of noisy data. The controllers estimated using both the model-free and model-based approaches provide similar closed-loop performances with 3.5 and 2.7% losses respectively, compared to a perfect controller that uses the true dynamic model and noise covariances of the HVAC system. Finally, we give future work directions for the model-free controller design approaches by discussing some remaining advantages of the model-based approaches.

1. Introduction

The development of a feedback controller for multivariable processes is typically accomplished by first estimating a dynamic model of the process from data, then solving an optimal control problem in real time for the process operation. This model-based controller design approach, known as model predictive control (MPC), is employed by practitioners in a range of industries (Darby and Nikolaou, 2012; Lahiri, 2017; Drgoňa et al., 2020; Vazquez et al., 2014). The theoretical stability and robustness properties of the MPC approach have also been established in the literature over the years (Mayne et al., 2000; de Oliveira and Biegler, 1994; Zheng and Morari, 1995), which make the technology reliable for implementation in industrial processes.

Reinforcement learning (RL) algorithms that avoid the dynamic model identification step have gained significant attention recently for model-free controller design as an alternative to the MPC approach. This interest is in part due to the impressive results of the model-free RL algorithms for playing Go (Silver et al., 2016) and Atari games (Mnih et al., 2015), and effectively performing robotic tasks (Kober et al., 2013; Levine and Koltun, 2014; Levine et al., 2016). The applications of model-free RL algorithms for process control (Shin et al., 2019; Nian et al., 2020; Jiang et al., 2021; Raman et al., 2020; Buşoniu et al., 2018), as well as other subfields of process systems engineering such as scheduling (Hubbs et al., 2020) and real time optimization (Powell et al., 2020) are currently active research topics.

For a controller design algorithm to be deployable in process control applications, the algorithm must be suitable to estimate a controller from a data set that contains both process and measurement noise. The algorithm should also require only a reasonable amount of training data. To the best of our knowledge, no previous work in the literature has demonstrated the effectiveness of a model-free RL algorithm to give a reasonable feedback controller from a data set that contains both process and measurement noise from a reasonable amount of training data. Standard model-based controller design approaches conveniently handle the noise sources in the dynamic model identification step. Hence, the research in the area of model-free RL must demonstrate the same capability for this class of algorithms to be competitive for consideration in an industrial implementation.

The goal of this paper is to build on the previous research in the area of model-free RL and present an approach that can handle both process and measurement noise and obtain a suitable controller.

* Corresponding author.
E-mail addresses: pratyushkumar@ucsb.edu (P. Kumar), jbraw@ucsb.edu (J.B. Rawlings).

https://doi.org/10.1016/j.compchemeng.2023.108325
Received 30 November 2022; Received in revised form 27 May 2023; Accepted 5 June 2023
Available online 14 June 2023

We first discuss the different types of model-free RL algorithms and previous work on their applications in feedback control.

1.1. Related work

Model-free RL algorithms have been studied in the machine learning field for decades using Markov decision processes as the underlying system. The text Sutton and Barto (2018) provides an introduction to the modern RL theory and applications. The model-free RL algorithms for decision-making and control purposes can be categorized into value function and policy gradient based methods. The value function based methods use the data collected from the system and an appropriate Bellman equation to estimate a state or state–action value function. The estimated function is then used to compute the optimal feedback policy or the control input. The feedback policy is typically found using an iterative approach known as policy iteration. The state–action value function is also called the Q-function, and the feedback policy in the Q-learning algorithms (Watkins and Dayan, 1992) is determined using estimates of this value function. The policy gradient methods (Williams, 1992; Silver et al., 2014; Lillicrap et al., 2015) first parameterize the feedback policy using a function approximator. Then, an optimization problem that maximizes the state value function is solved using an approximate gradient ascent approach to determine the parameters in the feedback policy.

Several approaches to use the model-free RL algorithms for feedback control problems have been proposed in the literature. Bradtke et al. (1994) presented a Q-learning algorithm to estimate the nominal linear quadratic regulator (LQR). This work presented a policy iteration algorithm based on successive Q-function approximations, and established the convergence of the iteration scheme to the optimal LQR feedback law. Lewis and Vamvoudakis (2011) and Rizvi and Lin (2017) treat the output measurement case in the Q-learning algorithm to estimate the nominal LQR. These two papers use a recent set of past measurements and control inputs as the state in the algorithm. All the above works require that the data set is generated from a deterministic linear system. In Rawlings and Maravelias (2019), we highlighted that the Q-learning algorithm to estimate the nominal LQR feedback law does not handle noise in the training data. This limitation renders the algorithm not suitable for implementation in applications.

Tu and Recht (2018) presented an algorithm to estimate the stochastic LQR feedback law obtained from the infinite horizon expected value objective (Bertsekas, 1995). The least squares policy iteration (LSPI) approach of Lagoudakis and Parr (2003) is applied, and linear systems with full state measurements and Gaussian process noise of *known* covariance are considered. The authors in Tu and Recht (2018) also present sample complexity analysis to estimate the LQR feedback law. And a rigorous theoretical comparison of the sample complexity requirements of a model-based and model-free approach to estimate the state value function in the LQR setting is provided in Tu and Recht (2019). These works aim to provide valuable insight about the data requirements of the LSPI algorithm to estimate the LQR feedback law, and show that it requires more data samples than the model-based approaches. However, the model-free algorithms proposed and analyzed in those papers are also not suitable for an industrial implementation because both process and measurement noise are present in applications, and the covariances of the noise statistics are almost always *unknown*. In this paper, we propose a systematic extension of the LSPI algorithm to make it more suitable for a practical implementation.

Policy gradient based algorithms to estimate the LQR feedback law have also been proposed in the literature (Fazel et al., 2018; Hambly et al., 2021). At each iteration, these algorithms compute the approximate gradient of the value function based on the data collected by implementing the current feedback law in the iteration to the system. This type of a model-free RL algorithm is called an *on-policy* algorithm, which can be impractical for an industrial implementation because a new data set should be generated for executing each iteration of the algorithm. Additionally, the entire data collection and controller estimation process should be repeated even for changes in the control problem tuning parameters. This repetition would be required because the tuning parameters change the value function used in the RL algorithm, and consecutively affect the feedback laws generated for the iterations.

The LSPI algorithm to estimate the stochastic LQR feedback law can also be implemented in an *offline, off-policy* approach (Lagoudakis and Parr, 2003; Krauth et al., 2019). Here, control input sequences independent of the feedback laws generated during the iterations can be used to collect the training data. The algorithm is implementable using a *fixed batch* (Lange et al., 2012) of data. And as long as the data is collected from a distribution spanning enough state and control input space, the algorithm is suitable to give a reasonable feedback controller. The fixed batch of data can be used for estimating the Q-functions and executing each iteration of the algorithm. This type of a model-free algorithm can have an improved data efficiency and is more suitable for an industrial implementation.

Yaghmaie et al. (2022) propose two off-policy algorithms to estimate the LQR feedback law for linear systems with state measurements and both process and measurement noise. Both the proposed algorithms are implemented in a *growing batch* approach (Lange et al., 2012). At each iteration of the algorithms, a new data set is collected by implementing the current feedback law in the iteration with an added probing noise. These algorithms can also require infeasible amounts of data for an industrial implementation, and a repetition of the data collection would be needed for a change in the LQR tuning parameters. By contrast, the LSPI algorithm for the LQR can be implemented offline using a fixed batch of data, which can be reused at each iteration of the algorithm. A change in the LQR tuning parameters requires only re-implementing the algorithm with the same data. Additionally, the work in Yaghmaie et al. (2022) treats only linear systems with state measurements and noise. In this paper, we demonstrate that our proposed modified LSPI algorithm can also be applied to linear systems with output measurements and both process and measurement noise.

There is also a considerable interest in the literature to apply *deep* RL algorithms for the control of nonlinear systems. The term "deep" is used when a NN is employed to represent the value function or the feedback policy. Spielberg et al. (2019) discuss the application of a deep RL algorithm to learn feedback controllers for nonlinear chemical engineering processes. Noise is added to the training data, but the case studies in that paper indicate that the algorithm can require infeasible amounts of data for applications. Wang et al. (2018) applies an actor–critic based reinforcement learning algorithm for nonlinear processes. The work in Yoo et al. (2021) uses a Monte-Carlo simulation based policy gradient algorithm to control batch processes. Dogru et al. (2021) demonstrate an application of a RL algorithm using experimental data collected from a three tank process. The authors first estimate a process model from data, which is then used to train an RL agent through offline simulations. The trained controller is applied to the real process and fine-tuned online, and the robustness of the final controller is also tested with measurement noise.

Other researchers have proposed approaches to treat constraints in the RL algorithms, improving other aspects of the control system such as controller tuning, and merging the RL and MPC approaches. Bangi and Kwon (2021) apply the deep deterministic policy gradient algorithm to a hydraulic fracturing process, and input constraints are treated using a modified reward function. The work in Pan et al. (2021) presents a constrained model-free Q-learning algorithm to treat probabilistic state constraints. Dogru et al. (2022) propose to use the deep RL approach for the tuning of proportional–integral controllers in industrial processes. The papers Morinelly and Ydstie (2016) and Zanon and Gros (2020) propose algorithms to combine the advantages of both the model-free RL and MPC approaches.

1.2. Contributions

In this paper, we develop a model-free RL algorithm that can be a viable option to estimate linear, unconstrained feedback controllers in process control applications as an alternative to the model-based approach. We study the problem of estimating the LQR feedback law and focus on using the Q-learning based least squares policy iteration (LSPI) algorithm because it can be implemented in an offline, off-policy approach.

First, we extend the least squares policy iteration (LSPI) algorithm proposed by Tu and Recht (2018) for linear systems with full state measurements driven by Gaussian process noise of known covariance to handle the case of an unknown noise covariance. We treat this case by modifying the function approximation architecture used in Tu and Recht (2018) to estimate the Q-function. Second, we demonstrate that the modified LSPI algorithm can be used to estimate a feedback controller for linear systems with only output measurements and data containing both process and measurement noise. A vector containing a recent set of past measurements and control inputs is used as a surrogate state in the LSPI algorithm. For comparison, we also discuss a maximum likelihood estimation (MLE) method that simultaneously estimates a linear dynamic model and noise covariances from data, which are used to develop a model-based feedback controller. We choose this system identification approach because it also estimates the noise covariances from data in addition to the linear dynamic model. The noise covariances are used to develop a Kalman filter for state estimation in the final closed-loop implementation.

Finally, we present illustrative simulation studies using an example system to compare the performances of the model-based and model-free controller design methods. We demonstrate that the proposed Q-learning algorithm in this paper can be a feasible option to estimate a feedback controller for the class of linear systems considered in this paper. The remaining advantages of the model-based approaches and some future work directions for the model-free RL algorithms to enable these latter class of algorithms competitive for consideration in industrial applications are discussed in the conclusions.

Outline. The rest of this paper is organized as follows. In the next section, we discuss the LQR formulation considered for controller design. We present the proposed Q-learning algorithm in Section 3. The MLE method to estimate a linear dynamic model and noise covariances is discussed in Section 4. Simulation studies to compare the Q-learning and MLE based controller design algorithms are presented in Section 5. Finally, we give the concluding remarks and future work directions in Section 6.

Notation. The notation $\mathbb{R}^n$ denotes a vector of real numbers in n dimensions. Bold symbol $\mathbf{d}$ denotes a time sequence, and a vector $d(k)$ denotes an element of $\mathbf{d}$ at time $k \geq 0$. The Kronecker product is denoted by $\otimes$. We use $\text{vec}(M)$ to represent the vector obtained by a vertical concatenation of all the columns in the matrix M. The identity matrix of size $n \times n$ is denoted by I_n. The Frobenius norm of a matrix M is denoted by $|M|$. We use $\text{diag}(v)$ to denote a diagonal matrix containing the elements of the vector v on the diagonal. The notation $\mathcal{N}(m, \Sigma)$ denotes a normal distribution of mean m and covariance Σ.

2. Linear quadratic regulator

The LQR formulation used for the controller design is described in this section. We consider linear systems driven by Gaussian process noise in the problem formulation. The objective function is the expected value of an infinite horizon sum of discounted, quadratic stage costs. The following optimization problem is solved to determine the LQR feedback law

$$\min_{\mathbf{u}} \mathbb{E}_{\mathbf{w}}\left[\sum_{k=0}^{\infty} \gamma^k (x(k)'Qx(k) + u(k)'Ru(k) + 2x(k)'Mu(k))\right]$$
$$\text{subject to} \quad x^+ = Ax + Bu + w, \quad w \sim N(0, Q_w) \tag{1}$$
$$x(0) = x \tag{2}$$

in which, $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^m$ is the control input, and $w \in \mathbb{R}^n$ is the Gaussian process noise. The matrices A and B characterize the linear dynamic model. The process noise has mean zero and covariance Q_w. The matrices Q, R, and M are used to penalize the state, control input, and the cross term in the stage cost. The discount factor ($0 < \gamma < 1$) is used in the problem formulation to ensure that the objective function is finite for any stabilizing feedback law. The expectation is performed over the noise sequence $\mathbf{w}$, and the decision variable is the control input sequence $\mathbf{u}$. The optimization problem is typically solved using a dynamic programming (DP) approach (Bertsekas, 1995, Pages 150-152), which gives the solution

$$u^0(k) = K^\star x^0(k) \tag{3}$$

Here, $u^0(k)$ and $x^0(k)$ are the optimal control input and state at the time step k. The matrix $K^\star$ is the optimal LQR feedback law that is determined by solving the following equations

$$\Pi^\star = Q + \gamma A'\Pi^\star A - (M + \gamma A'\Pi^\star B)(\gamma B'\Pi^\star B + R)^{-1}(M' + \gamma B'\Pi^\star A) \tag{4}$$
$$K^\star = -(R + \gamma B'\Pi^\star B)^{-1}(\gamma B'\Pi^\star A + M') \tag{5}$$

The first equation is a type of a discrete algebraic Riccati equation (DARE). The goal of both the model-based and model-free controller estimation algorithms is to determine the optimal feedback law $K^\star$ from data. We also define the value function corresponding to any stabilizing feedback law K as follows

$$V_K(x) = \mathbb{E}_{\mathbf{w}}\left[\sum_{k=0}^{\infty} \gamma^k \ell(x(k), Kx(k))\right] = x'\Pi x + \eta\,\text{tr}(\Pi Q_w) \tag{6}$$

in which $\ell(\cdot)$ is the quadratic stage cost (without the discount factor) in the optimization problem (1), and $\eta = \gamma/(1-\gamma)$. The value function quantifies the infinite horizon objective when control inputs according to the feedback law K, starting from the state x are implemented to the linear system. The cost-to-go matrix Π in the above value function is obtained by solving the following Lyapunov equation

$$\Pi = Q + K'RK + MK + K'M' + \gamma(A + BK)'\Pi(A + BK) \tag{7}$$

We may be interested in including a rate-of-change penalty on the control input in the stage cost with a matrix S_R as follows: $(u(k) - u(k-1))'S_R(u(k) - u(k-1))$. To include this penalty, we first augment the system state $x(k)$ with the previous control input $u(k-1)$ (Rao and Rawlings, 1999). Then, we construct the linear dynamic model and control problem penalty matrices for this augmented state such that the final LQR problem is of the type (1). Finally, the DARE Eq. (4) is solved for the LQR problem with the augmented state to determine the optimal feedback law.

3. Controller estimation using Q-learning

In this section, we first discuss the Q-learning based LSPI algorithm presented in Tu and Recht (2018) to estimate the optimal LQR feedback law obtained by solving the problem (1). The algorithm proposed in that work is not directly suitable for an implementation in industrial applications, because that paper assumes that the process noise covariance in the stochastic linear system is known. We propose the modifications that can be made to that algorithm to estimate a controller for linear systems with only output measurements and both process and measurement noise.

In the LSPI algorithms, we assume that the plant is a linear system of the type $x^+ = Ax + Bu + w$. This linear system with the stage cost $\ell(x, u)$ and the discount factor γ can be viewed as a Markov decision process $\mathcal{M} = (S, \mathcal{A}, \mathcal{P}, r, \gamma)$. Here, $S = \mathbb{R}^n$ is the space of states x and $\mathcal{A} = \mathbb{R}^m$ is the space of control inputs u. The function $\mathcal{P}(x, u, x^+)$ characterizes the probability of transitioning to the state x^+ from the

state x after taking the control input u. This probability can be obtained using the normal distribution $\mathcal{N}(Ax+Bu, Q_w)$. The reward function is $r = -\ell(x,u)$. The model-free RL literature typically uses a Q-function defined using the future rewards and performs a maximization over that function to derive the feedback policy. For the discussion below, we equivalently consider a Q-function defined using the future stage costs and perform a minimization over that function to determine the LQR feedback law.

3.1. Least squares policy iteration

The algorithm presented in Tu and Recht (2018) estimates the optimal LQR feedback law ($K^\star$) using a policy iteration approach. Each iteration of the algorithm consists a policy evaluation and an improvement step. In the policy evaluation step, the Q-function corresponding to the current feedback law in the iteration is estimated from data using a Bellman equation. Then, the feedback law for the next iteration is obtained in the policy improvement step. This policy iteration scheme is initialized with a stabilizing feedback law and the iterations are repeated until convergence of the feedback law. Next, we derive the Q-function for the LQR problem (1) and describe the policy iteration algorithm. The Q-function equations derived below differ slightly from Tu and Recht (2018) because we consider also the cross term in the LQR problem, and any general positive semidefinite process noise covariance.

The Q-function for any state x, control input u, and feedback law K_i is defined as

$$Q_{K_i}(x,u) = \ell(x,u) + \mathbb{E}_{\mathbf{w}}\left[\sum_{k=1}^{\infty} \gamma^k \ell(x(k), K_i x(k))\right] \tag{8}$$

We use the subscript i to denote the current iteration number in the algorithm. The Q-function is equal to the expected infinite horizon cost when the linear system ($x^+ = Ax+Bu+w$) is initialized with the state x, the control input u is implemented at the first time step, and all the subsequent control inputs are implemented based on the feedback law K_i. The second term in (8) is equal to the discount factor times an expectation of the value function corresponding to the state at the next time step x^+ and the feedback law K_i. Thus, the Q-function equation simplifies to

$$Q_{K_i}(x,u) = \ell(x,u) + \gamma \mathbb{E}_w[V_{K_i}(x^+)] \tag{9}$$

This expectation is evaluated over all the possible realizations of the process noise w at the first time step. The value function corresponding to the feedback law K_i is computed using (6), which is then substituted in the above equation. After evaluating the expectation over the noise w, we obtain the following structure for the Q-function

$$Q_{K_i}(x,u) = \begin{bmatrix} x \\ u \end{bmatrix}' \begin{bmatrix} Q+\gamma A'\Pi_i A & \gamma A'\Pi_i B + M \\ \gamma B'\Pi_i A + M' & R+\gamma B'\Pi_i B \end{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix} + \eta \,\mathrm{tr}(\Pi_i Q_w) \tag{10}$$

in which Π_i is the cost-to-go matrix in the value function corresponding to the feedback law K_i. In the subsequent discussions, we denote the matrix characterizing the quadratic term in this Q-function as

$$S_i = \begin{bmatrix} S_{ixx} & S_{ixu} \\ S_{iux} & S_{iuu} \end{bmatrix} = \begin{bmatrix} Q+\gamma A'\Pi_i A & \gamma A'\Pi_i B + M \\ \gamma B'\Pi_i A + M' & R+\gamma B'\Pi_i B \end{bmatrix} \tag{11}$$

We also note that the cost-to-go matrix Π_i is related to the Q-function matrix S_i by the equation

$$\Pi_i = \begin{bmatrix} I \\ K_i \end{bmatrix}' S_i \begin{bmatrix} I \\ K_i \end{bmatrix} \tag{12}$$

This relation can be verified by substituting for the matrix S_i from (11), and using the Lyapunov Eq. (7) for the feedback law and cost-to-go matrix pair (K_i, Π_i).

Policy evaluation. Based on the structural knowledge about the Q-function, the goal of the policy evaluation step is to approximate that function from process data. To achieve this goal, we use the following Bellman equation for the Q-function

$$Q_{K_i}(x,u) = \ell(x,u) + \gamma \mathbb{E}_w[Q_{K_i}(x^+, K_i x^+)] \tag{13}$$

Then, we parameterize the Q-function using a linear function approximation architecture, and formulate a least squares problem based on this Bellman equation to estimate the parameters in the function. The Bellman equation can also be derived by noting that the value function is related to the Q-function by $V_{K_i}(x) = Q_{K_i}(x, K_i x)$ and substituting this relation in (9). The following function approximation architecture is used to estimate the Q-function

$$Q_{K_i}(x,u) = \left[\begin{bmatrix} x \\ u \end{bmatrix}' \otimes \begin{bmatrix} x \\ u \end{bmatrix}' + \eta \mathrm{vec}(Q_w)' \left(\begin{bmatrix} I \\ K_i \end{bmatrix}' \otimes \begin{bmatrix} I \\ K_i \end{bmatrix}' \right) \right] \mathrm{vec}(S_i) \tag{14}$$

$$Q_{K_i}(x,u) = \phi(x,u)' \mathrm{svec}(S_i) \tag{15}$$

Eq. (14) is obtained by expanding the quadratic and trace terms in (10). Relation (12) is used to expand the trace term. The basis function $\phi(x,u)$ contains all the unique quadratic terms of the row vector in (14) containing the Kronecker products and the process noise covariance. The Q-function is linear in the parameters $\mathrm{svec}(S_i)$, which denotes a vector transformation of the symmetric matrix S_i containing all the diagonal elements and doubled off diagonal elements. The above Q-function approximation architecture is the same as used in the papers Tu and Recht (2017, 2018). Here, we present the architecture in a more general form for any positive definite noise covariance Q_w.

To estimate the Q-function from data, we first collect a training data trajectory by simulating the linear system $x^+ = Ax+Bu+w$ using some arbitrary control input and process noise sequences $\mathbf{u}$ and $\mathbf{w}$. This data generation step gives a corresponding state sequence $\mathbf{x}$. The following least squares problem is solved to estimate the parameters in the Q-function

$$\tilde{A}_i \ \mathrm{svec}(S_i) = \tilde{b}_i \tag{16}$$

The regressor ($\tilde{A}_i$) and target ($\tilde{b}_i$) matrices in this problem are constructed using the collected training data based on the following expressions

$$\tilde{A}_i = \frac{1}{N_t} \sum_{k=0}^{N_t} \phi(x(k),u(k)) \Big(\phi(x(k),u(k))' - \gamma \phi(x(k+1), K_i x(k+1))' \Big) \tag{17}$$

$$\tilde{b}_i = \frac{1}{N_t} \sum_{k=0}^{N_t} \phi(x(k),u(k)) \ell(x(k),u(k)) \tag{18}$$

in which, $x(k)$ and $u(k)$ are the state and control input at time step k in the training data trajectory, and N_t is the number of time steps in the trajectory. Lagoudakis and Parr (2003) provide a detailed derivation of the above matrices for the Q-learning least squares problem. At a high-level, the least squares matrices are obtained by enforcing the Q-function approximation to be a fixed point of the Bellman Eq. (13) in the space of representable functions by the linear approximation architecture in (15). The Q-function parameters estimated using the above least squares problem converge to the true parameters in the limit of large number of samples. This convergence property and the treatment of the noise in the least squares problem is an initial step towards developing an industrially implementable algorithm. We show in Sections 3.2–3.3 how to extend the algorithm so that it is suitable for applications.

Policy improvement. After the estimation of the parameters in the Q-function, the goal of the policy improvement step is to obtain an improved feedback law that has a smaller infinite horizon cost than the current feedback law in the iteration. This improved feedback law (Tu and Recht, 2018; Bradtke et al., 1994) is defined as

$$K_{i+1}x = \min_u \hat{Q}_{K_i}(x,u), \ \forall x \in \mathbb{R}^n \tag{19}$$

This step yields the following feedback law for the next iteration in the algorithm

$$K_{i+1} = -\hat{S}_{iuu}^{-1} \hat{S}_{iux} \tag{20}$$

in which, $\hat{S}_i$ is the estimate of the Q-function matrix obtained by solving the least squares problem (16) and transforming the vector of the parameter estimate back to the symmetric matrix form. The matrices $\hat{S}_{iuu}$ and $\hat{S}_{iux}$ are obtained using a block partition of the estimated Q-function matrix $\hat{S}_i$ similar to (11).

The policy evaluation step and improvement step described above are repeated for some specified number of iterations or until the estimated feedback law stops changing significantly. For the case studies in Tu and Recht (2018), the data were collected from the linear system in an episodic approach in which the initial states of the linear system are sampled from a specified distribution. We emphasize that the LSPI algorithm does not require the training data to be generated in an episodic approach, and the data can also be collected without any control on the initial state.

3.2. Q-function approximation under an unknown process noise covariance

The LSPI algorithm described in the previous subsection requires that the process noise covariance Q_w is known, because a value of this covariance must be specified in (15) to formulate the Q-learning least squares problem. In industrial applications, however, the noise covariance is typically unknown and the algorithms available in the literature to estimate noise covariances (Odelson et al., 2003) require that a linear dynamic model is available. Therefore, a direct application of the algorithm proposed in Tu and Recht (2018) is not suitable for an implementation in process control applications. In this subsection, we discuss how to handle an unknown covariance in the LSPI algorithm for linear systems of the type $x^+ = Ax+Bu+w$ with full state measurements and Gaussian process noise.

To estimate the Q-function when the process noise covariance (Q_w) is unknown, we propose to modify the linear function approximation architecture as follows

$$Q_{K_i}(x,u) = \begin{bmatrix} \begin{bmatrix} x \\ u \end{bmatrix}' \otimes \begin{bmatrix} x \\ u \end{bmatrix}', 1 \end{bmatrix} \begin{bmatrix} \mathrm{vec}(S_i) \\ \eta \,\mathrm{tr}(\Pi_i Q_w) \end{bmatrix} \tag{21}$$

$$Q_{K_i}(x,u) = \phi(x,u)' \begin{bmatrix} \mathrm{svec}(S_i) \\ \eta \,\mathrm{tr}(\Pi_i Q_w) \end{bmatrix} \tag{22}$$

in which, the basis function $\phi(x,u)$ is a vector that contains all the unique quadratic terms in the Kronecker product shown in (21), and concatenated with the scalar unity. The LSPI algorithm with this modified Q-function approximation architecture can be implemented in a similar approach as the algorithm in the previous subsection. The data generation and policy improvement steps remain the same. The only required change is in the policy evaluation step, in which a modified least squares problem (16) is solved and the matrices $\tilde{A}_i$ and $\tilde{b}_i$ are constructed using the basis function in (22).

In comparison with the algorithm in the previous section, this modified LSPI algorithm estimates an extra parameter $\eta \,\mathrm{tr}(\Pi_i Q_w)$ in the least squares problem. The extra parameter corresponds to the contribution of the process noise to the Q-function. We refer to that noise parameter as β in the rest of this paper. This parameter is not used directly in the policy improvement step, but an estimation of that parameter is required to systematically estimate the Q-function matrix S_i that is then used for the policy improvement.

We also note that the alternative of using a regularization approach in the least-squares problem in the LSPI algorithm does not suffice for adequately addressing the unknown process noise.[1] The modification of the Q-function architecture systematically retains the effect of the process noise in both the regressor and target matrices in the least squares problem. We show via simulation studies in Section 5 that the modification is suitable to provide convergence of the estimated controllers to the optimal LQR feedback law.

[1] We thank an anonymous reviewer for suggesting that we investigate this alternative approach.

3.3. Extension to output measurements and both process and measurement noise

The two algorithms described in the previous two subsections assume that full state measurements are available, and the linear system is affected by only process noise. In industrial systems, the sensor noise also contributes noticeably to the measurement and only some linear combination of the states may be measured. For these reasons, we wish to develop a feedback controller for the following class of linear systems

$$x^+ = Ax + Bu + w, \tag{23}$$

$$y = Cx + v, \tag{24}$$

$$w \sim N(0, Q_w), \quad v \sim N(0, R_v) \tag{25}$$

in which the matrix $C \in \mathbb{R}^{p\times n}$ characterizes the measurements $y \in \mathbb{R}^p$ that are some linear combinations of the states (x), v is the measurement noise, and R_v is the covariance of the measurement noise. To treat this output measurement case, we use a vector containing a recent set of past measurements and control inputs as a surrogate state. Then, we apply the LSPI algorithm from the previous subsection that does not assumes a known value of the noise covariance. For every time step k, the surrogate state is defined as

$$z(k) = \left[y(k-N_p)', \ldots, y(k-1)', u(k-N_p)', \ldots, u(k-1)' \right]' \tag{26}$$

in which, N_p is the number of past observations of measurements and control inputs used to construct the state. This approach of using a recent history of past observations as a surrogate state has been used for many years in both the system identification (Ho and Kalman, 1966; Qin, 2006) and Q-learning (Lewis and Vamvoudakis, 2011; Rizvi and Lin, 2017) fields for model and controller identification purposes. In this paper, we apply the approach in the LSPI algorithm to estimate a model-free feedback controller from noisy process data. The parameter N_p used to construct the surrogate state should be chosen to be a small value, but large enough such that the state contains sufficient information about the plant dynamics. We propose an approach to select the parameter N_p by examining condition numbers of a data matrix constructed using samples of the surrogate state. The approach is described in Appendix A.1.

For the case of output measurements, we consider an output tracking penalty ($y(k)'Q_y y(k)$) in the stage cost of the LQR problem. We use the data collected from a linear system of the type in Eqs. (23)–(24). The measurements of the surrogate state ($z(k)$), and the stage cost ($\ell(\cdot)$) values that use an output tracking penalty are used to construct the matrices (17)–(18) for the Q-learning least squares problem. We use the basis function in (22) that does not assume a known value of the noise covariance. The policy evaluation and improvement steps are implemented similarly as the LSPI algorithm in the previous subsections. The algorithm finally gives a feedback controller that uses the surrogate state (z) to compute the control input (u) during the closed-loop implementation. The main steps involved in the implementation of all the three model-free LSPI algorithms discussed in this section are summarized in Table 1.

When implementing the LSPI algorithm using the surrogate state, we assume that the plant is a linear system of the type $z^+ = A_z z + B_z u + w_z$, $y = C_z z$, for some unknown model matrices and covariance driving the process noise. The use of the surrogate state results in a dynamic and noise model *mismatch* between the plant (23)–(24) and the model assumed in the algorithm. We illustrate, however, in the simulation studies in Section 5 that this model mismatch is not crucial for a realistic linear system and noise covariances. The proposed LSPI algorithm for the output measurement case using the surrogate state is still suitable to give a reasonable feedback controller.

All the LSPI algorithms discussed in this paper estimate only a regulator and not a state estimator that can be used for noise filtering during the final closed-loop controller implementation. One of the advantages of the standard model-based algorithms is that since they also

Table 1
Summary of the main steps in the implementations of the model-free LSPI algorithms to estimate linear, unconstrained feedback controllers from noisy data.

State measurements with process noise and known covariance Q_w
• Collect training data $(\mathbf{u}, \mathbf{x})$ from the plant. • Initialize a stabilizing feedback law (K_1) and choose a number of iterations N_{itr} for the algorithm. For $i \in [1, N_{itr}]$, do • Construct the least squares matrices $(\tilde{A}_i, \tilde{b}_i)$ using the Q-function approximation architecture (14)–(15), Eqs. (17)–(18), and the collected training data. • Solve the least squares problem $\tilde{A}_i \text{svec}(S_i) = \tilde{b}_i$ to estimate the Q-function. • Get the feedback law for the next iteration K_{i+1} using (20).
State measurements with process noise and unknown covariance Q_w
• Perform the same steps as the LSPI algorithm with the known noise covariance, except (i) use the modified Q-function approximation architecture in (21)–(22) to construct the matrices $\tilde{A}_i, \tilde{b}_i$, and (ii) solve the least squares problem $\tilde{A}_i \begin{bmatrix} \text{svec}(S_i) \\ \beta_i \end{bmatrix} = \tilde{b}_i$ to estimate the Q-function.
Output measurement case with both process and measurement noise
• Collect training data $(\mathbf{u}, \mathbf{y})$ from the plant, and construct time series of the surrogate state $(\mathbf{z})$ using the definition (26). • Initialize a stabilizing feedback law that acts on the surrogate state. Perform the same steps as the LSPI algorithm above that does not assume a known noise covariance, using samples of the surrogate state (z) to construct the least squares matrices $(\tilde{A}_i, \tilde{b}_i)$.

estimate a dynamic model (possibly also noise covariances), a Kalman filter can then be constructed using the estimated model and noise covariances to optimally perform state estimation during the closed-loop implementation. Since the model-free Q-learning algorithms lose the ability to develop an optimal state estimator, we use a heuristic noise filtering approach; we deploy a sample exponential moving average of the measurements for the closed-loop simulation studies. Thus, the filtered estimates of the measurements are obtained using

$$\hat{y}^+ = \alpha \hat{y} + (1 - \alpha) y^+ \tag{27}$$

in which, $\hat{y}$ and $\hat{y}^+$ are the filtered measurements at the current and the next time step respectively, y^+ is the measurement at the next time step, and α is a tuning parameter that characterizes the sensitivity of the estimate to the new measurement. During the closed-loop implementation, the filtered measurement obtained using this approach are used to construct the surrogate state z that is then used to compute the control input. In the simulation studies, we also consider a case in which a linear system of the type $x^+ = Ax + Bu + w$ with full state measurements and only Gaussian process noise is used as the plant. In this case, we perform noise filtering on the state measurements and use the filtered states to compute the control inputs during the closed-loop implementation.

4. Maximum likelihood estimation of linear dynamic model and noise covariances

In the simulation studies in this paper, we aim to compare the performances of the LSPI algorithms discussed above with a model-based algorithm that handles the noisy data in the identification step. We next present a maximum likelihood estimation (MLE) algorithm to estimate a linear dynamic model and noise covariances from process data. The estimated linear model is used to develop a regulator, while both the model and noise covariances are used to construct a Kalman filter for state estimation. The MLE algorithm discussed below is similar to the Larimore's subspace method (Qin, 2006; Larimore, 1990), and this algorithm was also recently discussed in Kuntz and Rawlings (2022).

To obtain the MLE estimates of the linear dynamic model and noise covariances, we assume that a linear system of the type (23)–(24) is used to generate the training data. Then, we set up the following equation for each time step k

$$s(k) = \Theta t(k) + e(k), \quad e(k) \sim N(0, S_{wv}) \tag{28}$$

in which, $s = \begin{bmatrix} x^+ \\ y \end{bmatrix}$, $t = \begin{bmatrix} x \\ u \end{bmatrix}$, $e = \begin{bmatrix} w \\ v \end{bmatrix}$, and $\Theta = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$. The MLE estimates can be derived by maximizing the probability of the observed measurements (Graham and Rawlings, 2022, Section 4.7) over the parameters Θ and S_{wv}. These estimates can then be computed using the following two equations

$$\hat{\Theta} = \Big(\sum_{k=0}^{N_s-1} s(k) t(k)' \Big) \Big(\sum_{k=0}^{N_s-1} t(k) t(k)' \Big)^{-1} \tag{29}$$

$$\hat{S}_{wv} = \frac{1}{N_s - (n+m)} \sum_{k=0}^{N_s-1} (s(k) - \hat{\Theta} t(k))(s(k) - \hat{\Theta} t(k))' \tag{30}$$

Here, N_s denotes the number of training data samples. Since the state x for the linear system (23)–(24) is not measured, we use the surrogate state defined in (26) to construct samples of the vectors s and t as follows

$$s = \begin{bmatrix} z^+ \\ y \end{bmatrix}, \quad t = \begin{bmatrix} z \\ u \end{bmatrix} \tag{31}$$

This method is similar to the approach used for the output measurement case in the LSPI algorithm discussed previously. The parameter N_p to construct the surrogate state can be chosen similarly as the LSPI algorithm using the approach described in Appendix A.1. For the simulation case in which all the states are measured and the plant is a linear system driven by only process noise, we use $s = x^+$ and $t = [x, u]'$. The MLE algorithm for this case estimates only the linear model matrices A and B, and the process noise covariance Q_w.

5. Simulation studies

In this section, simulation studies are presented with a linear heating, ventilation, and air-conditioning (HVAC) system to demonstrate that the proposed LSPI algorithm in this paper can be successfully used to estimate a feedback controller from noisy process data. We present simulation studies with two cases of the plant used for the data generation and closed-loop simulations. First, we consider the full state measurement case in which the plant is a linear system of the type $x^+ = Ax + Bu + w$, driven by only process noise. This case is considered to examine the performances of the LSPI algorithms when there is no mismatch between the data generating system and the model assumed in the algorithm. Second, we consider the output measurement case in which the HVAC plant is a linear system of the type (23)–(24). For both of these cases, we compare the data requirements of the model-based and the model-free algorithms to obtain a reasonable controller. Additionally, we examine the closed-loop performances of the estimated controllers compared to a perfect model-based controller that uses the actual plant model and noise covariances.

The HVAC building system shown in Fig. 1 is considered for the simulation studies. The building system has two zones, which give four states corresponding to the zone and mass temperatures ($x = [T_{z_1}, T_{m_1}, T_{z_2}, T_{m_2}]'$) of each zone. The control inputs are the cooling duty ($u = [\dot{Q}_{cz_1}, \dot{Q}_{cz_2}]'$) to each zone, and the disturbances to the system are the heat disturbance to each zone and the ambient

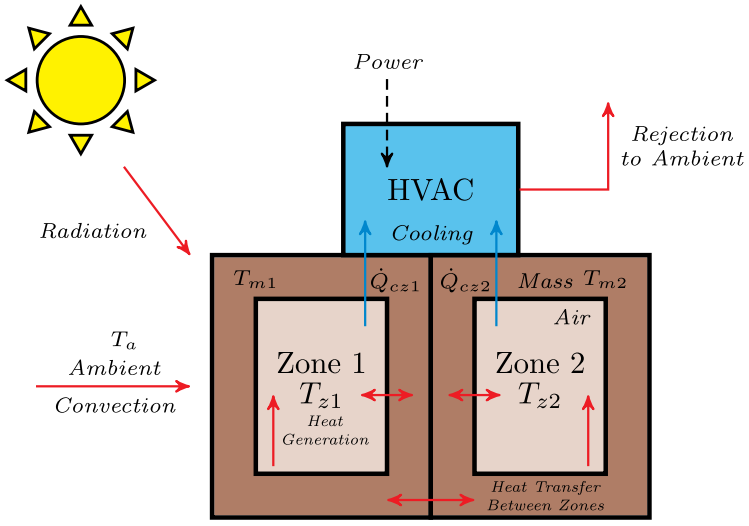

Fig. 1. A diagram of the HVAC building system considered for simulation studies.

Table 2
Parameters used in the ODEs of the HVAC building system, and the steady state used to obtain deviation variables. The ambient temperature does not interact with the mass in the model so H_{m_1} and H_{m_2} are zero. The interaction coefficients (β_{ij}) for the combination of temperature states not shown in this Table are zero.

Parameter	Value	Parameter	Value
H_{z_1}/C_{z_1}	6×10^{-4} s^{-1}	H_{z_2}/C_{z_2}	1×10^{-3} s^{-1}
$\beta_{z_1-m_2}/C_{z1}$	2×10^{-4} s^{-1}	$\beta_{z_2-m_2}/C_{z_2}$	5×10^{-4} s^{-1}
$\beta_{z_1-m_2}/C_{m1}$	3×10^{-4} s^{-1}	$\beta_{m_1-m_2}/C_{m_2}$	2×10^{-4} s^{-1}
$\beta_{m_1-m_2}/C_{m1}$	6×10^{-4} s^{-1}	$\beta_{z_2-m_2}/C_{m_2}$	4×10^{-4} s^{-1}
$1/C_{z_1}$	2.5×10^{-4} °C/kJ	$1/C_{z_2}$	2×10^{-4} °C/kJ
u_s = [50 kW, 40 kW]'			
p_s = [60 kW, 40 kW, 22 °C]'			
x_s = [25.53 °C, 23.65 °C, 22.23 °C, 22.70 °C]'			

temperature ($p = \left[\dot{Q}_{az_1},\ \dot{Q}_{az_2},\ T_a\right]'$). The linear ordinary differential equations (ODEs) shown in (32) are considered for the building system. We convert the ODEs to a linear discrete time state space model to simulate the plant. For the full state measurement case, we measure all the four states. While for the output measurement case, the zone temperatures ($y = \left[T_{z_1},\ T_{z_2}\right]'$) are the only measurements. We assume that the disturbances to the building system remain constant, and all the variables in the simulations are considered in deviation from a fixed steady-state. The sample time for the measurements is chosen as 1 min. The parameters used for the ODEs in (32), and the steady state used to obtain deviation variables are given in Table 2. Code for all the simulation studies presented in this paper is available at: https://github.com/pratyushkumar211/qlearning_noisy_data_2023.

$$C_i\frac{dT_i}{dt} = -H_i(T_i - T_a) - \sum_{j\neq i}\beta_{i-j}(T_i - T_j) - \dot{Q}_{ci} + \dot{Q}_{ai} \tag{32}$$
$$(i,\ j) \in \{z_1, m_1, z_2, m_2\}$$

5.1. State measurements with process noise

We first examine the performances of the LSPI and MLE algorithms for the full state measurement case with only process noise. To generate the training data, we simulate the plant using Gaussian random control input and process noise sequences starting from the origin as the initial state. The mean of both these sequences are zero, and the covariances are given in Table 3. We generate a total of 60 h (3600 samples) of training data.

After the data generation step, we implement three controller estimation algorithms, (i) SYSID: The MLE algorithm discussed in Section 4

Table 3
Covariances used to generate the training data and the LQR problem tuning parameters used for the simulation studies. The matrix Σ_u denotes the covariance used to generate the control input signal. The matrices Q_s, Q_{ys}, and R_s are diagonal containing the inverse of the squares of the state, measurement, and control input at the steady state shown in Table 2.

Parameter	Value
State measurement with process noise	
Σ_u	$400I_2$
Q_w	diag([0.08, 0.01, 0.09, 0.01]')
Q	10^3Q_s
R	10^2R_s
S_R	10^2R_s
γ	0.98
Output measurement with process and measurement noise	
Σ_u	$625I_2$
Q_w	diag([0.08, 0.01, 0.09, 0.01]')
R_v	diag([0.3, 0.2]')
Q_y	10^3Q_{ys}
R	10^2R_s
S_R	10^2R_s
γ	0.98

is used to estimate a linear dynamic model (A, B) and the process noise covariance (Q_w). These estimates are then used to develop a LQR feedback law for regulation and a Kalman filter for state estimation, (ii) LSPI-KQW: The LSPI algorithm in Section 3.1 that assumes a known value of the process noise covariance is implemented to estimate the optimal LQR feedback law, and (iii) LSPI-UQW: The LSPI algorithm in Section 3.2 that considers no knowledge of the process noise covariance, and also estimates the noise contribution (β) to the Q-function is implemented. In the LSPI-KQW algorithm, we use the actual noise covariance of the plant that was used in the data generation step. A comparison between the LSPI-KQW and LSPI-UQW algorithms illustrates the achievable performance with an LSPI algorithm that does not assumes any knowledge of the process noise statistics.

We implement the three algorithms discussed above with varying amounts of training data starting from 3 and up to 60 h, in increments of 3 h. The following stage cost is considered in the LQR problem

$$\ell(x, u, \Delta u) = x'Qx + u'Ru + \Delta u'S_R\Delta u \tag{33}$$

The penalty matrices and the discount factor are given in Table 3. To implement the rate-of-change penalty on the control input in the LSPI algorithms, we use samples of the augmented state $\tilde{x} = \left[x',\ \ u'_{-1}\right]'$ containing one previous control input in the least squares problems.

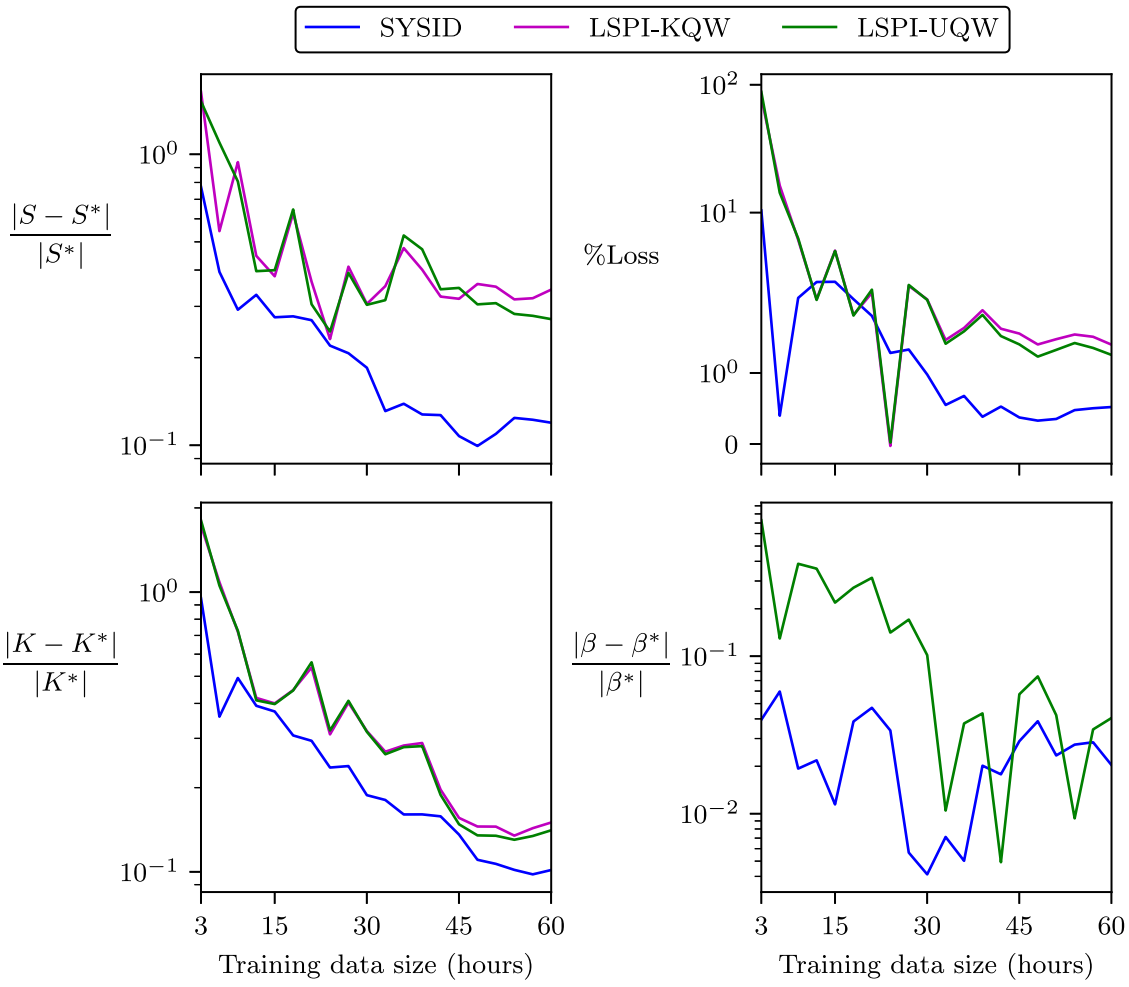

Fig. 2. The accuracy of the Q-function and LQR feedback law estimates obtained using the SYSID, LSPI-KQW, and LSPI-UQW algorithms with varying amounts of training data for the full state measurement case. We also show the variation in the % closed-loop performance loss (top right) compared to a perfect model-based controller that uses the true plant model and noise covariance.

The augmented state gives a Q-function matrix S of dimensions 8×8, and the LQR feedback law is of size 2×6. The initial stabilizing feedback law for the two LSPI algorithms is chosen as a matrix of zeros, and we implement the policy evaluation and improvement steps in the algorithms for a total of 15 iterations.

First, we analyze the estimates of the optimal Q-function and LQR feedback law obtained with the three algorithms discussed above for varying amounts of training data. To examine the quality of the Q-function estimates, we compare the estimated S matrix and the noise term β with the corresponding quantities computed using the true plant model. For the SYSID algorithm, the Q-function and LQR feedback law estimates are computed using the estimated dynamic model and noise covariance. Fig. 2 shows the variations in the errors in the estimated Q-functions and feedback laws for all the three algorithms obtained with varying amounts of training data. We observe that for a small amount of 3 h of data, the errors in the optimal S matrix, noise term β, and feedback law K for the two LSPI algorithms are unacceptably large, around 60–110%. But the estimation errors in all the three quantities improve with increasing amount of training data. The errors in the estimated LQR feedback law with the LSPI algorithms using the full 60 h of training data are between 10–20%. All the estimated quantities obtained with the SYSID algorithm are consistently similar or better than the two LSPI algorithms. We also observe that the performances of the LSPI-KQW and LSPI-UQW algorithms are approximately similar. This observation demonstrates that the proposed approach of using a modified Q-function approximation architecture to treat an unknown process noise covariance is effective. The approach provides a similar performance as the impractical LSPI-KQW that uses the true plant noise covariance.

Second, we compare the closed-loop performances of all the estimated controllers with a perfect model-based controller that uses the true plant model and noise covariance. For these comparisons, we conduct multiple closed-loop simulations with each estimated controller starting from some random initial states of the plant. All these initial states in each closed-loop simulation are sampled randomly from a uniform distribution in the range -10 to 10 °C. We use $\alpha = 0.2$ for noise filtering in (27) during the closed-loop implementation of the model-free controllers. Based on all the closed-loop simulations, we compute the following metric representing the overall performance of each estimated controller

$$\Lambda_T^C = \frac{1}{N_{tr} N_t} \sum_{j=1}^{N_{tr}} \sum_{k=0}^{N_t} \ell(x_j(k), u_j(k), \Delta u_j(k)) \tag{34}$$

in which, the subscript j denotes the trajectory number of a closed-loop simulation, N_{tr} is the total number of simulations, and N_t is the number of time steps in each simulation. We perform $N_{tr} = 500$ simulations, each for $N_t = 240$ (4 h) time steps. This length of the closed-loop simulations is chosen such that we capture enough transient and steady-state data in the performance metrics. For each estimated controller, we further compute the following loss metric that compares the performance with the perfect model-based controller

$$\% \text{ Performance Loss} = 100(\lambda_T^C - \Lambda_T^P)/\Lambda_T^P \tag{35}$$

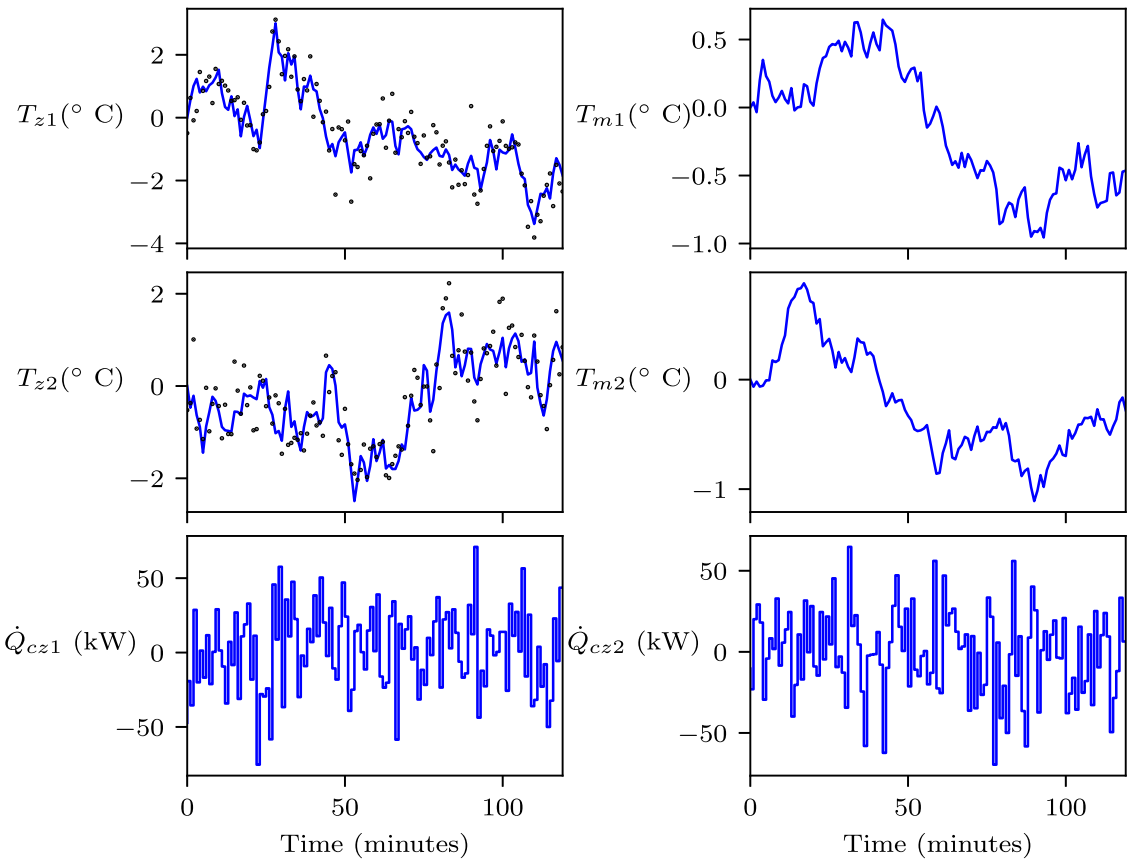

Fig. 3. Sample training data used for the controller estimation in the LSPI and SYSID algorithms for the output measurement case. The black dots in the zone temperature plots show the measurements while the blue lines depict the actual state.

Here, Λ_T^P denotes the performance of the perfect model-based controller that uses the true plant model and noise covariance. In Fig. 2 (top right), we also show the variation in the above performance loss metric for the estimated controllers with the three algorithms for varying amounts of training data. We observe that after around 15 h of data, the performances of the controllers estimated using the two LSPI algorithms are close to the perfect controller with the loss metric less than 5%. The performances of all the estimated controllers further improve with an increase in the amount of training data.

Table 4 summarizes the loss metric obtained by the controllers estimated with the three algorithms when using the full 60 h of training data. The model-based controller obtained using the SYSID algorithm has the best performance with a 0.52% loss. However, the two model-free controllers also have a reasonably well performance with only 1.4% and 1.25% loss metrics. The better performance of the model-based controller estimated using the SYSID algorithm is because the estimated LQR feedback law is more accurate as we observe in Fig. 2. We also perform state estimation during the closed-loop implementation of that controller using a Kalman filter developed using the estimated model and noise covariance.

We also report the controller estimation times by the three algorithms in Table 4 when using the entire 60 h of training data. The SYSID algorithm uses the least amount of time of less than 1 s, whereas, the two LSPI algorithms use around 6.9 and 12 s to estimate the LQR feedback law. The LSPI algorithms solve multiple least squares problems in an iterative approach, which requires more time for the controller estimation. The larger estimation time for the LSPI-KQW compared to the LSPI-UQW is due to the fact that the Q-function approximation architecture used to construct the least squares problem matrices in the former algorithm has more Kronecker product evaluations. The controller estimation times of both LSPI algorithms are reasonable, however. The two algorithms can be conveniently implemented offline in industrial applications to estimate a feedback controller after the data collection step from the plant.

Table 4
Metrics summarizing the closed-loop performance of the controllers estimated using the SYSID and LSPI algorithms. We also show the offline controller estimation times of the algorithms.

Controller	% Performance Loss	Controller estimation time (seconds)
State measurement with process noise		
SYSID	0.52	0.05
LSPI-KQW	1.4	11.94
LSPI-UQW	1.25	6.88
Output measurement with process and measurement noise		
SYSID	2.68	0.03
LSPI	3.55	3.15

5.2. Output measurements with process and measurement noise

We now analyze the performances of the LSPI and MLE algorithms for the output measurement case with both process and measurement noise. We generate the training data by simulating the plant using Gaussian random control input, process and measurement noise sequences starting from the origin as the initial state. The mean of all these sequences are zero, and the covariances are given in Table 3. We generate a total of 24 h (1440 samples) of training data. Fig. 3 shows the first 2 h of the generated training data.

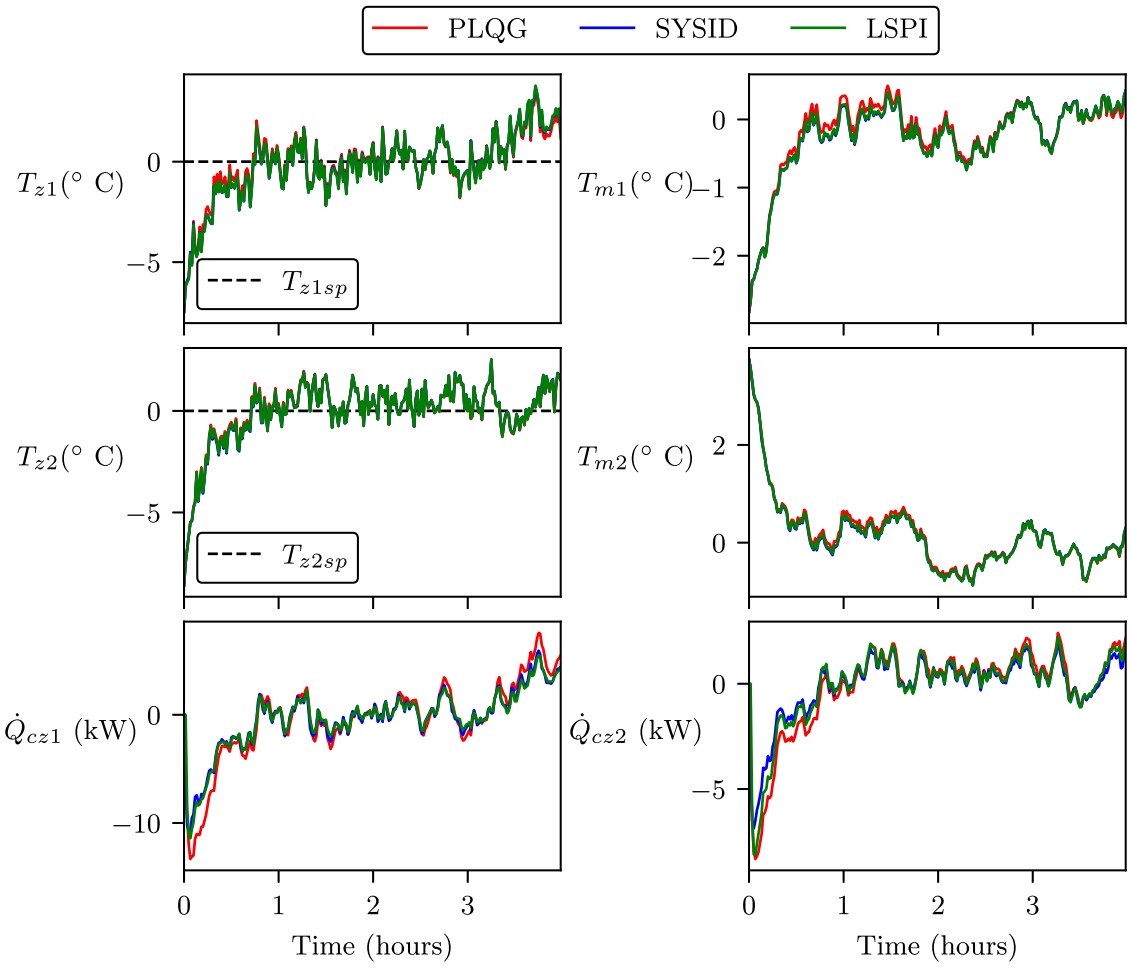

Fig. 4. A sample closed-loop simulation using the controllers estimated with the SYSID and LSPI algorithms, and the perfect PLQG controller. The performance of the controller estimated using the LSPI algorithm is close to the other two model-based controllers.

After the data generation step, we implement two algorithms to estimate a feedback controller, (i) SYSID: The MLE algorithm discussed in Section 4 is implemented to estimate a linear dynamic model (A, B, C, D) and noise covariances (S_{wv}), which are then used to develop a regulator and state estimator, and (ii) LSPI: The model-free algorithm for the output measurement case discussed in Section 3.3 is implemented to estimate a feedback controller. For both these two algorithms, we use a surrogate state containing $N_p = 2$ number of past measurements and control inputs after examining the condition numbers of the data matrix described in Appendix A.1. For the LSPI algorithm, we use a matrix of zeros as the initial stabilizing feedback law and implement the policy evaluation and improvement steps for a total of 15 iterations.

We consider an LQR problem with an output tracking penalty in the stage cost as follows

$$\ell(y, u, \Delta u) = y' Q_y y + u' R u + \Delta u' S_R \Delta u \tag{36}$$

The penalty matrices and the discount factor considered for the problem are given in Table 3. The dimension of the surrogate state (z) is 8, which requires the estimation of a Q-function matrix S of size 10×10, and a feedback law of dimensions 2×8. As mentioned previously, the use of the surrogate state creates a mismatch between the plant and the model assumed in both the controller estimation algorithms. The estimated Q-function and feedback law cannot be directly compared to some true quantities. Therefore, we compare only the closed-loop performances of the estimated controllers with a perfect model-based controller that uses the true plant model and noise covariances, which is subsequently referred as the PLQG controller.

First, we examine the closed-loop performances of the estimated controllers using the two algorithms compared to the PLQG controller. For this analysis, we implement the controller estimation algorithms using the full 24 h of data. Then, we perform multiple closed-loop simulations using all the three controllers starting from some random initial states. All of the initial states in these simulations are sampled from a uniform distribution in the range -10 to 10 °C. We perform 500 of such closed-loop simulations, each for a length of 4 h. We use $\alpha = 0.2$ for noise filtering in (27) during the implementation of the model-free controller estimated using the LSPI algorithm. In Fig. 4, we show one of the closed-loop trajectories obtained using the PLQG and the controllers estimated using the SYSID and LSPI algorithms. We observe that the performance of the controller estimated using the LSPI algorithm is similar to the performances of the PLQG and the controller estimated using the SYSID method. For each closed-loop simulation, we also compute the following performance metric

$$\Lambda_{Tj}^{C} = \frac{1}{N_t} \sum_{k=0}^{N_t} \ell(y_j(k), u_j(k), \Delta u_j(k)) \tag{37}$$

in which, the subscript j denotes the trajectory number of a closed-loop simulation. In Fig. 5 (left), we show the histograms of the above performance metric obtained from the closed-loop simulations performed with all the three controllers. The histograms illustrate that the performance of the controller estimated using the LSPI algorithm is almost the same as the other two model-based controllers across all the simulations.

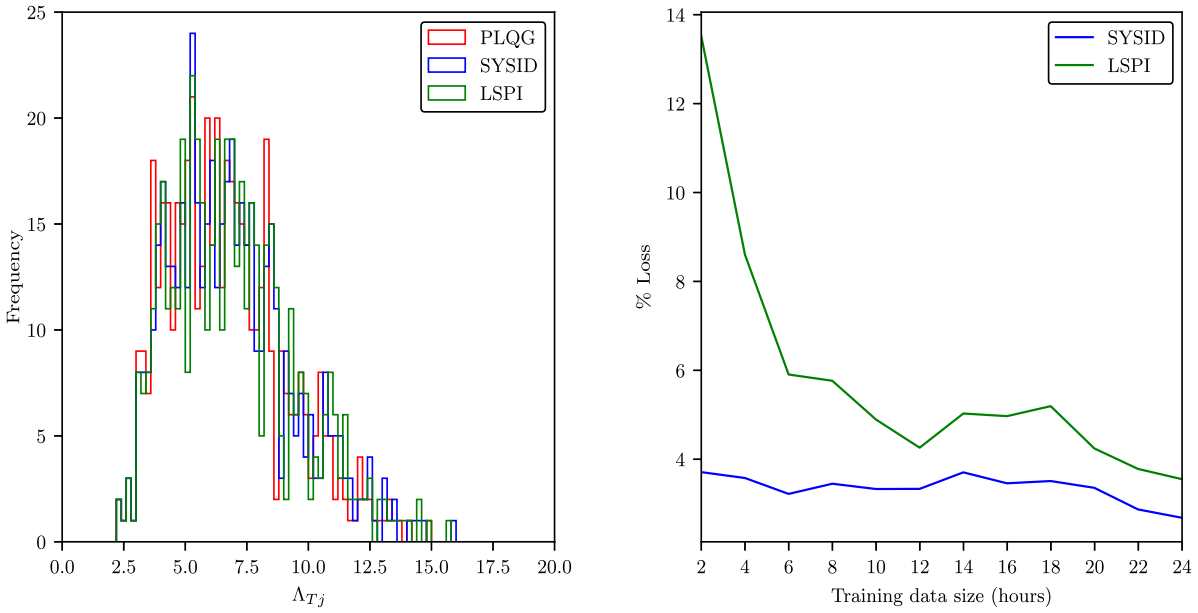

Fig. 5. Metrics summarizing the closed-loop performances of the controllers estimated using the SYSID and LSPI algorithms. Left plot shows a histogram of the performance metrics for multiple closed-loop simulations performed with the controllers estimated using the full 24 h of training data. The right plot shows the variation in the performance loss by the controllers estimated using the two algorithms compared to the PLQG for varying amounts of training data.

Further, we study the data efficiency of the SYSID and LSPI algorithms to obtain a reasonable controller. For this analysis, we implement the SYSID and LSPI algorithms to estimate a feedback controller with varying amounts of training data starting from 2 and up to 24 h, in increments of 2 h. With each estimated controller, we perform the multiple closed-loop simulations starting from some random initial states as performed for the previous analysis. Then, we compute the loss metric shown in (35) that captures the overall performance of an estimated controller compared to the PLQG. Fig. 5 (right) shows the variations in the loss metric for the SYSID and LSPI algorithms with varying amounts of training data. For a small amount of 2 h of training data, the performance of the controller estimated using the LSPI algorithm is noticeably worse with around 14% performance loss, but the performances of the controllers estimated using the LSPI algorithm improve with an increase in the training data. The SYSID algorithm is more data efficient compared to the LSPI method, and the performance loss is less than 4% with only 2 h of training data.

In Table 4, we show the loss metric compared to the PLQG controller and the controller estimation times for the two algorithms when using the entire 24 h of training data. The controller estimated using the SYSID algorithm provides a performance loss of 2.68%, while the LSPI controller also provides a good performance with a loss metric of only 3.55% compared to the PLQG controller. The offline controller estimation times for the SYSID and LSPI algorithms are 0.02 and 3.2 s, which shows that both the algorithms are suitable for an implementation in industrial applications.

6. Conclusions and discussion

In this paper, a novel model-free controller estimation approach using Q-learning and least squares policy iteration (LSPI) has been presented. The approach is suitable to estimate an accurate controller for linear systems from data sets containing both process and measurement noise, and such a model-free RL algorithm has not been previously presented in the literature. We first extended an available LSPI algorithm in the literature by modifying the Q-function approximation architecture to treat an unknown process noise covariance. Then, we proposed to use a surrogate state containing a recent history of past measurements and control inputs to treat the case of output measurements with both process and measurement noise in the modified LSPI algorithm.

We demonstrated via simulation studies that the proposed LSPI algorithm in this paper is suitable to obtain a feedback controller that subsequently has a reasonable closed-loop performance. For the output measurement case, we illustrated that the controller estimated using the model-free LSPI algorithm has only a 3.55% performance loss compared to a perfect model-based controller that uses the true plant model and noise covariances. The controller estimated using the MLE algorithm has a better performance with a loss metric of 2.68%. The data required to obtain the above performance with the LSPI algorithm was 24 h. These observations highlight that the proposed model-free controller estimation approach using least squares policy iteration can also be a feasible option to estimate an unconstrained feedback controller for industrial plants that have approximately linear dynamics. The algorithm can be applied if the disturbances to the plant remain constant or change in small magnitudes during the training data collection.

The reasons for the success of the LSPI algorithm proposed in this paper compared to the failure of the algorithm in Bradtke et al. (1994) as pointed out in Rawlings and Maravelias (2019) are two-fold. First, the algorithm in this paper is developed from Tu and Recht (2018) that uses a stochastic LQR problem formulation and the plant is a linear system driven by an explicit process noise term. The Q-function structure and the least-squares problem solution accounts for the stochastic linear system. The algorithm in Bradtke et al. (1994) assumes that the linear plant under consideration evolves nominally without any stochastic noise term. The treatment of the noise by Tu and Recht (2018) enables some robustness of the algorithm to noisy data sets; however, it cannot be applied directly because the algorithm in that paper assumes that the covariance of the process noise is known. Second, we modified the linear Q-function approximation architecture to handle an unknown process noise covariance. The modified algorithm was then used to estimate a feedback law for the output measurement case with both process and measurement noise.

Despite the advancement in the area of model-free RL presented in this paper, however, some more issues should be addressed in future research to enable an industrial deployment of these class of algorithms.

For a controller design algorithm to be implementable in applications, the algorithm must have some more features apart from being suitable to estimate the controller from noisy data. The final control algorithm should achieve offset-free closed-loop performance on the primary variables, a systematic method should be used to perform noise filtering on the measurements, and process constraints on the control inputs and measurements should be accounted during the controller design and final deployment steps. The model-based MPC control algorithm has the capability to systematically account for all these features that led to a widespread adoption of the approach by practitioners in a range of industries. We next describe how the above features are included in the model-based algorithm and outline future work directions for the model-free approaches.

- Industrial processes are typically affected by nonzero mean unmeasured disturbances, and it is often required to maintain some primary variables at their setpoints in the presence of such disturbances to achieve a satisfactory control performance. Integrating disturbance models can be used in a model-based controller to achieve a zero setpoint tracking error in the primary variables. The issue of achieving offset-free performance with a model-free RL based feedback controller should be addressed in future research.
- An estimated dynamic model and noise covariances of the process can be used to develop a Kalman filter to systematically perform state estimation during the closed-loop implementation. This state estimation approach cannot be implemented with a model-free controller. Thus, an appropriate noise filtering approach for an implementation with a model-free RL controller should be developed.
- Lastly, industrial systems often have actuator and measurement constraints that can be naturally incorporated in an online MPC optimization problem when using a dynamic model of the process. Future research should also be directed towards developing approaches to estimate a constrained feedback controller from noisy process data.

The model-free controller design approach presented in this paper is a step towards an industrially implementable algorithm. The model-based approaches still have the advantages discussed above. However, if future advancements in the model-free RL approach can address these issues, this class of methods may prove to be suitable for industrial applications.

Declaration of competing interest

The authors Pratyush Kumar and James B. Rawlings declare no potential competing interests.

Data availability

Data will be made available on request.

Acknowledgments

The funding for this work was provided by the industrial members of the Texas-Wisconsin-California Control Consortium (TWCCC).

Appendix

A.1. Model order selection

The parameter N_p used to construct the surrogate state (z) in the LSPI and MLE algorithms for the output measurement case determines the assumed model order of the plant. The parameter can be selected as follows. First, we construct the data matrix

$$H = \left[z(N_p),\ z(N_p+1),\ \ldots,\ z(N_t)\right] \tag{38}$$

in which, N_t is the total number of time steps in the training data. As shown in (26), each column of this data matrix contains a vector of N_p past inputs and outputs. If we choose N_p too large, i.e., model order too large, then the rows of the data matrix become nearly linearly dependent. The condition number of the data matrix becomes large, and the estimates of the dynamic model or the Q-function parameters become sensitive to noise in the data since most past measurements and control inputs do not provide additional information about the process dynamics. If we choose N_p too small (model order too small), then an accurate dynamic model or Q-function cannot be obtained from the data. So the tradeoff is accuracy of the estimates versus sensitivity to noise. To evaluate the tradeoff, we examine the condition number of the data matrix for different values of N_p to gauge when additional past measurements and control inputs stop providing further information about the process dynamics. We increase N_p from unity to a large integer, and plot the value of the data matrix condition number. We select the value of N_p just before the "knee" in this curve where condition number increases rapidly and then saturates as N_p increases further. A more rigorous method to choose the model order for the model-free LSPI algorithm can be developed in a future work.

References

Bangi, M.S.F., Kwon, J.S.-I., 2021. Deep reinforcement learning control of hydraulic fracturing. Comput. Chem. Eng. 154, 107489.
Bertsekas, D.P., 1995. Dynamic Programming and Optimal Control, Volume 2. Athena Scientific, Belmont, MA.
Bradtke, S.J., Ydstie, B.E., Barto, A.G., 1994. Adaptive linear quadratic control using policy iteration. In: American Control Conference, 1994, Volume 3. IEEE, pp. 3475–3479.
Buşoniu, L., de Bruin, T., Tolić, D., Kober, J., Palunko, I., 2018. Reinforcement learning for control: Performance, stability, and deep approximators. Annu. Rev. Control 46, 8–28.
Darby, M.L., Nikolaou, M., 2012. MPC: Current practice and challenges. Control Eng. Pract. 20 (4), 328–342.
de Oliveira, N.M.C., Biegler, L.T., 1994. Constraint handling and stability properties of model-predictive control. AIChE J. 40 (7), 1138–1155.
Dogru, O., Velswamy, K., Ibrahim, F., Wu, Y., Sundaramoorthy, A.S., Huang, B., Xu, S., Nixon, M., Bell, N., 2022. Reinforcement learning approach to autonomous PID tuning. Comput. Chem. Eng. 161, 107760.
Dogru, O., Wieczorek, N., Velswamy, K., Ibrahim, F., Huang, B., 2021. Online reinforcement learning for a continuous space system with experimental validation. J. Proc. Cont. 104, 86–100.
Drgoňa, J., Arroyo, J., Figueroa, I.C., Blum, D., Arendt, K., Kim, D., Ollé, E.P., Oravec, J., Wetter, M., Vrabie, D.L., et al., 2020. All you need to know about model predictive control for buildings. Annu. Rev. Control 50, 190–232.
Fazel, M., Ge, R., Kakade, S., Mesbahi, M., 2018. Global convergence of policy gradient methods for the linear quadratic regulator. In: International Conference on Machine Learning. PMLR, pp. 1467–1476.
Graham, M.D., Rawlings, J.B., 2022. Modeling and Analysis Principles for Chemical and Biological Engineers, second ed., Paperback Nob Hill Publishing, Santa Barbara, CA, ISBN: 978-0-9759377-6-1, 560 pages.
Hambly, B., Xu, R., Yang, H., 2021. Policy gradient methods for the noisy linear quadratic regulator over a finite horizon. SIAM J. Cont. Opt. 59 (5), 3359–3391.
Ho, B.L., Kalman, R.E., 1966. Efficient construction of linear state variable models from input/output functions. Regelungstechnik 14, 545–548.
Hubbs, C.D., Li, C., Sahinidis, N.V., Grossmann, I.E., Wassick, J.M., 2020. A deep reinforcement learning approach for chemical production scheduling. Comput. Chem. Eng. 141, 106982.
Jiang, Z., Risbeck, M.J., Ramamurti, V., Murugesan, S., Amores, J., Zhang, C., Lee, Y.M., Drees, K.H., 2021. Building HVAC control with reinforcement learning for reduction of energy cost and demand charge. Energy Build. 239, 110833.
Kober, J., Bagnell, J.A., Peters, J., 2013. Reinforcement learning in robotics: A survey. Int. J. Robot. Res. 32 (11), 1238–1274.
Krauth, K., Tu, S., Recht, B., 2019. Finite-time analysis of approximate policy iteration for the linear quadratic regulator. Adv. Neural Inf. Process. Syst. 32.

Kuntz, S.J., Rawlings, J.B., 2022. Maximum likelihood estimation of linear disturbance models for offset-free model predictive control. In: American Control Conference. Atlanta, GA, pp. 3961–3966.

Lagoudakis, M.G., Parr, R., 2003. Least-squares policy iteration. J. Mach. Learn. Res. 4, 1107–1149.

Lahiri, S.K., 2017. Multivariable Predictive Control: Applications in Industry. John Wiley & Sons, Inc., Hoboken, NJ.

Lange, S., Gabel, T., Riedmiller, M., 2012. Batch reinforcement learning. In: Reinforcement Learning: State-of-the-Art. Springer, pp. 45–73.

Larimore, W.E., 1990. Canonical variate analysis in identification, filtering, and adaptive control. In: Proceedings of the 29th Conference on Decision and Control. pp. 596–604.

Levine, S., Finn, C., Darrell, T., Abbeel, P., 2016. End-to-end training of deep visuomotor policies. J. Mach. Learn. Res. 17 (1), 1334–1373.

Levine, S., Koltun, V., 2014. Learning complex neural network policies with trajectory optimization. In: Int. Conf. Mach. Learn.. PMLR, pp. 829–837.

Lewis, F.L., Vamvoudakis, K.G., 2011. Reinforcement learning for partially observable dynamic processes: Adaptive dynamic programming using measured output data. IEEE Trans. Syst. Man Cybern. B 41 (1), 14–25.

Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D., 2015. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.

Mayne, D.Q., Rawlings, J.B., Rao, C.V., Scokaert, P.O.M., 2000. Constrained model predictive control: Stability and optimality. Automatica 36 (6), 789–814.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al., 2015. Human-level control through deep reinforcement learning. Nature 518 (7540), 529.

Morinelly, J.E., Ydstie, B.E., 2016. Dual MPC with reinforcement learning. IFAC–P. Online 49 (7), 266–271.

Nian, R., Liu, J., Huang, B., 2020. A review on reinforcement learning: Introduction and applications in industrial process control. Comput. Chem. Eng. 139, 106886.

Odelson, B.J., Rajamani, M.R., Rawlings, J.B., 2003. A New Autocovariance Least-Squares Method for Estimating Noise Covariances. Technical Report 2003-04, TWMCC, Department of Chemical Engineering, University of Wisconsin–Madison.

Pan, E., Petsagkourakis, P., Mowbray, M., Zhang, D., del Rio-Chanona, E.A., 2021. Constrained model-free reinforcement learning for process optimization. Comput. Chem. Eng. 154, 107462.

Powell, K.M., Machalek, D., Quah, T., 2020. Real-time optimization using reinforcement learning. Comput. Chem. Eng. 143, 107077.

Qin, S.J., 2006. An overview of subspace identification. Comput. Chem. Eng. 30, 1502–1513.

Raman, N.S., Devraj, A.M., Barooah, P., Meyn, S.P., 2020. Reinforcement learning for control of building HVAC systems. In: Proc. Ame. Contro. Con.. IEEE, pp. 2326–2332.

Rao, C.V., Rawlings, J.B., 1999. Steady states and constraints in model predictive control. AIChE J. 45 (6), 1266–1278.

Rawlings, J.B., Maravelias, C.T., 2019. Bringing new technologies and approaches to the operation and control of chemical process systems. AIChE J. 65 (6), http://dx.doi.org/10.1002/aic.16615.

Rizvi, S.A.A., Lin, Z., 2017. Output feedback reinforcement Q-learning control for the discrete-time linear quadratic regulator problem. In: Decision and Control (CDC), 2017 IEEE 56th Annual Conference on. IEEE, pp. 1311–1316.

Shin, J., Badgwell, T.A., Liu, K.-H., Lee, J.H., 2019. Reinforcement learning–overview of recent progress and implications for process control. Comput. Chem. Eng. 127, 282–294.

Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al., 2016. Mastering the game of Go with deep neural networks and tree search. Nature 529 (7587), 484–489.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., Riedmiller, M., 2014. Deterministic policy gradient algorithms. In: ICML.

Spielberg, S., Tulsyan, A., Lawrence, N.P., Loewen, P.D., Bhushan Gopaluni, R., 2019. Toward self-driving processes: A deep reinforcement learning approach to control. AIChE J. 65 (10), e16689.

Sutton, R.S., Barto, A.G., 2018. Reinforcement Learning: An Introduction. MIT Press.

Tu, S., Recht, B., 2017. Least-squares temporal difference learning for the linear quadratic regulator. arXiv preprint arXiv:1712.08642.

Tu, S., Recht, B., 2018. Least-squares temporal difference learning for the linear quadratic regulator. In: International Conference on Machine Learning. PMLR, pp. 5005–5014.

Tu, S., Recht, B., 2019. The gap between model-based and model-free methods on the linear quadratic regulator: An asymptotic viewpoint. In: Conference on Learning Theory. PMLR, pp. 3036–3083.

Vazquez, S., Leon, J.I., Franquelo, L.G., Rodriguez, J., Young, H.A., Marquez, A., Zanchetta, P., 2014. Model predictive control: A review of its applications in power electronics. IEEE Ind. Syst. Mag. 8 (1), 16–31.

Wang, Y., Velswamy, K., Huang, B., 2018. A novel approach to feedback control with deep reinforcement learning. IFAC–P. Online 51 (18), 31–36.

Watkins, C.J., Dayan, P., 1992. Q-learning. Mach. Learn. 8 (3–4), 279–292.

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

Yaghmaie, F.A., Gustafsson, F., Ljung, L., 2022. Linear quadratic control using model-free reinforcement learning. IEEE Trans. Autom. Cont..

Yoo, H., Kim, B., Kim, J.W., Lee, J.H., 2021. Reinforcement learning based optimal control of batch processes using Monte-Carlo deep deterministic policy gradient with phase segmentation. Comput. Chem. Eng. 144, 107133.

Zanon, M., Gros, S., 2020. Safe reinforcement learning using robust MPC. IEEE Trans. Autom. Cont. 66 (8), 3638–3652.

Zheng, A., Morari, M., 1995. Stability of model predictive control with mixed constraints. IEEE Trans. Autom. Cont. 40 (10), 1818–1823.