Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 55-12 (2022) 494–499

Parameter Optimal Iterative Learning Control Design: from Model-based, Data-driven to Reinforcement Learning [⋆]

Yueqing Zhang [*] Bing Chu [*] Zhan Shu [**]

[*] *University of Southampton, SO17 1BJ Southampton, SO17 1BJ, UK (e-mail: {yz3n17, b.chu}@soton.ac.uk).*

[**] *University of Alberta, Edmonton, T6G 2H5, Canada (e-mail: zshu1@ualberta.ca)*

Abstract: Iterative learning control (ILC) is a high-performance control design method for systems operating in a repetitive fashion by learning from past experience. Our recent work shows that reinforcement learning (RL) shares many features with ILC and thus opens the door to new ILC algorithm designs. This paper continues the research by considering a parameter optimal iterative learning control (POILC) algorithm. It has a very simple structure and appealing convergence properties, but requires a model of the system. We first develop a data-driven POILC algorithm without using model information by performing an extra experiment on the plant. We then use a policy gradient RL algorithm to design a new model-free POILC algorithm. Both algorithms achieve the high-performance control target without using model information, but the convergence properties do differ. In particular, by increasing the number of function approximators in the latter, the RL-based model-free ILC can approach the performance of the model-based POILC. A numerical study is presented to compare the performance of different approaches and demonstrate the effectiveness of the proposed designs.

Keywords: Iterative learning control, reinforcement learning control, data-based control.

1. INTRODUCTION

High-performance control systems working in a repetitive manner play a key role in a wide range of areas, e.g., manufacturing, health care, and robotics. For such problems, conventional control methods would have difficulties in achieving the high performance control targets, as they normally require a highly accurate model which can be expensive or even impossible to obtain.

To solve the above problem, iterative learning control (ILC) is proposed that enables high-performance tracking without an accurate model. ILC updates the input signal based on previous data (including input and error information) to meet the stringent requirements regarding the control accuracy, and thus avoids the use of accurate model information. This learning mechanism makes ILC efficient in diverse high-performance applications, including robotics (Norrlöf, 2002), jet print (Park et al., 2007), etc. A number of ILC methods have been proposed and can be divided into two categories. Model-based ILC design uses explicit system dynamics to design the input updating law, examples of which include gradient-based ILC (Owens et al., 2009), inverse-based ILC (Harte et al., 2005), norm optimal ILC (Amann et al., 1996), parameter optimal ILC (Owens and Feng, 2003), etc. On the other hand, model-free design (or data-driven design) directly updates the input without using model information, by either explicitly or implicitly identifying or adapting system (controller) parameters from the data or performing extra experiments on the plant (Janssens et al., 2012; Bolder et al., 2018). For more detailed review of ILC techniques, please refer to Bristow et al. (2006); Owens (2015). It is worth pointing out that, model-free ILC algorithms, which can achieve the high tracking performance without using model information, tend to converge slower than model-based ILC design. A new model-free ILC algorithm with comparable convergence as model-based algorithms has been waiting.

Recently, reinforcement learning (RL) has received much research interest (Bertsekas, 2019). RL learns the best policy (control) from continuous or repeated interactions with the environment to maximise a performance index (called return). In our earlier work (Zhang et al., 2019b), we show that RL shares many similarities with ILC and can be used to solve ILC problems. We show via simulation that RL-based ILC designs can achieve the high performance tracking requirement but tend to converge slower than model-based ILC, opening new possibilities for novel ILC algorithm design using advanced RL designs. Another work (Poot et al., 2020) proposes an actor-critic ILC method and compares it with a basis function approach in the norm optimal framework. It shows that it is capable of achieving the same feed-forward signal without using explicit model information. More recently, we develop a Q-learning based norm optimal ILC design (Zhang et al., 2022). We can show the convergence rigorously. However, it has a relatively complex structure, and needs a substantial number of iterations to converge.

[⋆] This work was partially supported by the ZZU-Southampton Collaborative Research Project 16306/01 and the China Scholarship Council (CSC).

This paper continues our previous work along the above research line by considering an alternative parameter optimal iterative learning control (POILC) algorithm. POILC has simple structure with appealing convergence properties. We will show that the model-based POILC design can be implemented without using any model information. First, a data-driven POILC using experiments on the plant is developed and its convergence properties are analysed. Next, policy gradient, a popular method in RL, is applied to develop an RL-based POILC which requires no model information or system identification procedure. The simulation shows that both proposed algorithms can achieve perfect tracking of the reference without using any model information, though the convergence properties do differ.

The rest of the paper is structured as follows: in Section 2, a high performance tracking problem is described and a model-based POILC algorithm is reviewed. In Section 3, a data-driven POILC algorithm is developed and its convergence properties are given. In Section 4, we draw upon tools from RL to develop another model-free POILC algorithm. Section 5 provides a simulation example to compare the performance of the proposed algorithms with model-based design. Finally, Section 6 concludes this paper and discusses the future research directions.

2. HIGH PERFORMANCE TRACKING CONTROL AND POILC

In this section, the design problem is formulated and a model-based POILC solution is reviewed.

2.1 High performance tracking using ILC

A discrete-time single-input-single-output (SISO) linear-time-invariant (LTI) system is considered in this paper with the following state-space representation

$$\begin{aligned} x_k(t+1) &= Ax_k(t) + Bu_k(t), \quad x_k(0) = x_0, \\ y_k(t) &= Cx_k(t), \quad t = 0, 1, ..., N \end{aligned} \tag{1}$$

where t is the time index and N is the trial length; k is the trial index; $x_k \in \mathbb{R}^n$, $u_k(\cdot) \in \mathbb{R}$ and $y_k(\cdot) \in \mathbb{R}$ are the state, input and output at trial k, where n is the system order; A, B and C are system matrices with appropriate dimensions. The system is required to track the same reference $r(t)$ defined on $t \in [0, N]$ with a high accuracy repeatedly, i.e., at $t = N+1$, the time is reset to $t = 0$ and the state is reset to the identical initial value x_0, the system then performs the same tracking task again.

Assuming the relative degree of the system is one, i.e., $CB \neq 0$, system (1) can be written into a lifted form as

$$y_k = Gu_k + d, \tag{2}$$

where the system matrix G and unforced response d are

$$G = \begin{bmatrix} CB & 0 & \dots & 0 \\ CAB & CB & \dots & 0 \\ \vdots & \vdots & \ddots & 0 \\ CA^{N-1}B & CA^{N-2}B & \dots & CB \end{bmatrix} \tag{3}$$

$$d = [CAx_0, CA^2x_0, \dots, CA^Nx_0]^T; \tag{4}$$

u_k and y_k are the super-vectors to represent the inputs and outputs at each time point on trial k, i.e.

$$u_k = [u_k(0), u_k(1), \dots, u_k(N-1)]^T, \tag{5}$$

$$y_k = [y_k(1), y_k(2), \dots, y_k(N)]^T. \tag{6}$$

Note that, systems with a higher relative degree can be treated similarly, please refer to Owens et al. (2009).

Without loss of generality, it can be assumed that $d = 0$ by incorporating it into the reference (i.e., replacing r by $r - d$), where r is the lifted representation of the reference signal, i.e., $r = [r(1), r(2), \dots, r(N)]^T$. Accordingly, the vector form of the error in the trial k is $e_k = r - y_k$.

ILC Design Problem: The ILC design problem can then be described as finding an updating law input

$$u_{k+1} = f(u_k, e_k) \tag{7}$$

such that the output follows the reference perfectly, i.e.

$$\lim_{k\to\infty} e_k = 0 \tag{8}$$

where f is a function of the previous trial's input and error.

2.2 Parameter optimal iterative learning control

A number of ILC design algorithms have been proposed in the literature to solve the above high performance tracking problem. In this paper, we consider the POILC proposed in Owens and Feng (2003) as shown below

$$u_{k+1} = u_k + \gamma_{k+1}e_k, \tag{9}$$

where the learning gain γ_{k+1} is chosen by minimising the following performance index

$$J_{k+1}(\gamma_{k+1}) := \|e_{k+1}\|^2 + \omega\gamma_{k+1}^2, \tag{10}$$

where $\omega > 0$ is a small weighting scalar on the learning gain. If the system model G is known, then the solution is

$$\gamma_{k+1}^* = \frac{e_k^T G e_k}{\omega + e_k^T G^T G e_k}. \tag{11}$$

The model-based POILC algorithm has some nice convergence properties, as summarised in the following theorem.

Theorem 1. (Owens and Feng (2003)) The POILC algorithm, defined by equations (9) and (11), achieves monotonic convergence in the tracking error norm, i.e.

$$\|e_{k+1}\| \leq \|e_k\|, \qquad k = 0, 1, 2, \dots$$

and the convergence in the learning gain to zero, i.e.

$$\lim_{k\to\infty} \gamma_{k+1}^* = 0.$$

Furthermore, if $G + G^T > 0$, then the perfect tracking is achieved, i.e., $\lim_{k\to\infty} e_k = 0$.

Note that, when the system model is known, the optimal control parameter (11) with good convergence properties can be calculated; however, when the system model is not available, the optimal parameter selection will be a challenge, which will be explored in the following sections.

3. DATA-DRIVEN BASED POILC

In this section, we will develop a data-driven approach to implement the POILC algorithm without using the model 'G'. This is achieved by implementing an extra experiment on the plant using error collected from the last ILC trial, and the details are given below.

Define $\bar{y}_k$ as follows

$$\bar{y}_k = Ge_k, \tag{12}$$

then the optimal learning gain γ_{k+1}^* can be calculated as

$$\gamma_{k+1}^* = \frac{e_k^T \bar{y}_k}{\omega + \bar{y}_k^T \bar{y}_k}. \tag{13}$$

Note that, $\bar{y}_k = Ge_k$ is the response of the system to input e_k and therefore can be obtained by carrying out an extra experiment on the plant. Note that, one optimal input update requires a normal ILC trial and an extra experiment. To avoid the confusion of the concept 'trial', the data-driven POILC can be restated as

$$u_k = \begin{cases} u_0, & k = 0 \\ e_{k-1}, & k = 1, 3, 5, \dots \\ u_{k-2} + \gamma_k e_{k-2}, & \text{otherwise} \end{cases} \tag{14}$$

and accordingly, the learning gain for the even trials is

$$\gamma_k = \frac{e_{k-2}^T y_{k-1}}{\omega + y_{k-1}^T y_{k-1}}, k = 2, 4, \dots \tag{15}$$

in which an even k represents an ILC trial and an odd k represents an experimental step.

The above control design can be summarised into a data-driven POILC algorithm, as shown in Algorithm 1.

Algorithm 1 Data-driven POILC

```
Input: Initial input $u_0$; maximum ILC trial number $k_{max}$
Output: Input for the next trial $u_{k+1}$
  Initialisation : Set $k = 0$
  for $k \leq k_{max}$ do
    if $k$ is even then
      Apply $u_k$ to the plant to collect $y_k, e_k$
    else
      Apply $e_k$ to the plant to collect $\bar{y}_k$
      Compute $\gamma_{k+1}$ according to (13)
      Update the input $u_{k+1}$ according to (14)
    end if
  end for
```

Theorem 2. The data-driven POILC Algorithm 1 achieves monotonic convergence in the tracking error norm for normal ILC trials, i.e.

$$\|e_{k+2}\| \leq \|e_k\|, \qquad k = 0, 2, 4, \dots \tag{16}$$

and the convergence in the learning gain to zero, i.e.

$$\lim_{k\to\infty} \gamma_{k+2}^* = 0.$$

Furthermore, if $G + G^T > 0$, perfect tracking of the reference can be achieved for all the ILC trials, i.e.

$$\lim_{k\to\infty} e_k = 0, \qquad k = 0, 2, 4, \dots \tag{17}$$

The detailed proof can be found in Appendix A.

Note that the proposed data-driven POILC algorithm achieves monotonic convergence of the tracking error norm without any prior of model information. This is achieved by conducting an extra experiment on the plant between two normal ILC trials to update the input for tracking, doubling the number of experiments required to achieve the same accuracy (as in the model-based design). Besides conducting extra experiments, another way to achieve model-free control is RL, as can be seen in the next section.

Remark 1. The idea of performing an experiment on the plant (instead of using the model) is not new. A similar idea has been used in, e.g. Owens et al. (2009) and Bolder et al. (2018) for gradient based algorithms (e.g. $G^T e_k$).

4. REINFORCEMENT LEARNING BASED POILC

In this section, methods from RL are used to develop a new model-free POILC algorithm. First, the POILC design is formulated into a Markov decision process (MDP), then the policy gradient method, a popular method in RL, is applied and the corresponding algorithm is presented.

4.1 MDP formulation of POILC design

POILC design can be reformulated into an MDP defined by a tuple (X, U, f, R), where

- X is the set of states. The state at the k^{th} trial is the error $e_k \in \mathbb{R}^N$ and N is the trial length.
- U is the set of actions. The action at the k^{th} trial is the learning gain $\gamma_{k+1} \in \mathbb{R}$.
- $f(e_k, \gamma_{k+1})$ is the transition function according to

$$e_{k+1} = r - y_{k+1} = (I - \gamma_{k+1} G)e_k. \tag{18}$$

- $R(e_k, \gamma_{k+1})$ is the reward defined as follows

$$\begin{aligned} R(e_k, \gamma_{k+1}) &= \|e_{k+1}\|^2 + \omega\gamma_{k+1}^2 \\ &= e_k^T (I - \gamma_{k+1} G^T)(I - \gamma_{k+1} G)e_k + \omega\gamma_{k+1}^2. \end{aligned} \tag{19}$$

The core of an MDP is to find a policy $\boldsymbol{\pi}$ to determine the action to take based on the current state, i.e., $\gamma_{k+1} = \boldsymbol{\pi}(e_k)$, to minimise the return defined as

$$\boldsymbol{R} = \sum_{k=0}^{\infty} \eta^k R(e_k, \gamma_{k+1}) \tag{20}$$

where $0 \leq \eta \leq 1$ is the discount factor which determines the effect of the future cost to the return, i.e., a larger value of η stands for more importance of the future cost. In the POILC problem, $\eta = 0$.

4.2 Policy gradient based POILC algorithm

Policy gradient method is a popular reinforcement learning technique that solves the problem by optimising parameterised policies in respect of maximising the expected return (or minimising the long-term cumulative cost in the control setting) by gradient descent method.

In this paper, since the action space is continuous, i.e., $\gamma_{k+1} \in \mathbb{R}$, it is assumed to be chosen from a normal distribution parameterised by $\theta \in \mathbb{R}^m$ as follows

$$\boldsymbol{\pi}(\gamma_{k+1}|e_k, \theta) = \frac{1}{\sqrt{2\pi}\sigma(e_k, \theta)} \exp\left(-\frac{[\gamma_{k+1} - \mu(e_k, \theta)]^2}{2\sigma(e_k, \theta)^2}\right), \tag{21}$$

where $\boldsymbol{\pi}$ is a parameterised policy; $\mu(e_k, \theta) : X \times \mathbb{R}^m \to \mathbb{R}$, is the mean of the policy $\boldsymbol{\pi}$; $\sigma(e_k, \theta) : X \times \mathbb{R}^m \to \mathbb{R}^+$, is the standard deviation of the parametric policy $\boldsymbol{\pi}$; θ is the policy's parameter vector that can be divided into two parts, i.e., $\theta = [\theta_\mu, \theta_\sigma]^T$, where θ_μ and θ_σ are parameters for the mean and standard deviation respectively.

For the stochastic policy in equation (21), the performance index can be written as an expectation, i.e.,

$$\boldsymbol{J}(\boldsymbol{\pi}(\gamma_{k+1}|e_k, \theta)) = \mathbb{E}_{\boldsymbol{\pi}}\{\boldsymbol{R}(\theta)\}, \tag{22}$$

where $\boldsymbol{J}(\boldsymbol{\pi}(\gamma_{k+1}|e_k, \theta))$ denotes the expected return and is abbreviated as $\boldsymbol{J}(\theta)$; $\mathbb{E}_\pi$ represents the expectation over the distribution $\pi(\gamma_{k+1}|e_k, \theta)$; $\boldsymbol{R}(\theta)$ is the return obtained by applying policy $\boldsymbol{\pi}(\gamma_{k+1}|e_k, \theta)$.

Policy gradient methods seek to minimise the performance index (22) by updating approximate gradient descent in the expected return $\boldsymbol{J}$ with respect to the policy's parameters. Given that the system (environment) is unknown, directly computing this gradient is a challenging task. The policy gradient theorem (Sutton and Barto, 2018) tackles with this problem by reformulating the derivative of the performance index (expected return)

$$\nabla_\theta\{\boldsymbol{J}(\theta)\}=\nabla_\theta\{\mathbb{E}_{\boldsymbol{\pi}}\{\boldsymbol{R}(\theta)\}\} = \mathbb{E}_{\boldsymbol{\pi}}\{\nabla_\theta(\ln(\boldsymbol{\pi}))\boldsymbol{R}(\theta)\} \quad (23)$$

where ∇_θ denotes the gradient with respect to θ.

A sample-based stochastic policy gradient method to update the policy's parameters is then given as follows

$$\begin{aligned}\theta_{k+1} &= \theta_k - \alpha_k \nabla_\theta \boldsymbol{J}(\theta_k)\\ &= \theta_k - \alpha_k \nabla_\theta \ln \boldsymbol{\pi}(\gamma|e,\theta)\boldsymbol{R}(\theta_k),\end{aligned} \quad (24)$$

where $\alpha_k > 0$ is the step size.

With the above, a policy gradient based POILC (PG-POILC) is given in Algorithm 2, and the convergence to a locally optimal policy can be proved in the Theorem 3, following the convergence analysis in Sutton et al. (2000) under mild conditions.

Algorithm 2 PG-POILC

```
Input: Initial input $u_0$ and parameter vector $\theta_0$; maximum ILC trial number $k_{max}$
Output: Input for the next trial $u_{k+1}$
1: Initialisation : Set $k = 0$
2: for $k = 0$ to $k_{max}$ do
3:     Input $u_k$ to the plant to collect the error $e_k$
4:     Compute $\mu(e_k, \theta_k)$ and $\sigma(e_k, \theta_k)$
5:     Select an action $\gamma_{k+1}$ according to equation (21)
6:     Update the input $u_{k+1} \leftarrow u_k + \gamma_{k+1}e_k$
7:     Update $\theta_{k+1}$ according to equation (24)
8: end for
```

Theorem 3. Suppose the reward function R, the policy $\boldsymbol{\pi}(\theta)$ and the learning step size sequence $\{\alpha_k\}_{k=0}^{\infty}$ in (24) satisfy the following conditions:

(1) The reward R is bounded for any state and action, i.e., $R(e,\gamma) \le \boldsymbol{B}_R$.
(2) The parameterised policy $\boldsymbol{\pi}(\theta)$ is a differentiable control policy with respect to θ, with $\nabla_\theta \ln \boldsymbol{\pi}(\gamma|e,\theta)$ to be bounded and $\boldsymbol{L}$-Lipschitz, i.e., $\|\nabla_\theta \ln \boldsymbol{\pi}(\gamma|e,\theta)\| \le \boldsymbol{B}$, $\|\nabla_\theta \ln \boldsymbol{\pi}(\gamma|e,\theta_p) - \nabla_\theta \ln \boldsymbol{\pi}(\gamma|e,\theta_q)\| \le \boldsymbol{L}\|\theta_p - \theta_q\|$, for some constants $\boldsymbol{B}$ and $\boldsymbol{L}$, for any e, γ, θ, θ_p, θ_q.
(3) The step size sequence $\{\alpha_k\}_{k=0}^{\infty}$ satisfies $\lim_{k\to\infty}\alpha_k = 0$, $\sum_{\infty}^{k=0}\alpha_k = \infty$ and $\sum_{\infty}^{k=0}\alpha_k^2 < \infty$.

Then, the sequence $\{\boldsymbol{J}(\theta_k)\}_{k=0}^{\infty}$ starting with any θ_0 and following the update law (24), converges such that $\lim_{k\to\infty}\frac{\partial \boldsymbol{J}(\theta_k)}{\partial\theta} = 0$.

Note that using policy gradient method, a policy is drawn every time from the normal distribution and then applied to the system. The 'stochasticity' here ensures that sufficient 'exploration' happens and therefore an 'optimal' solution can be found. In the following section, a method to parametrise the policy is provided as an example.

Remark 2. The boundedness of the reward in the condition (1) in Theorem 3 is a standard assumption in the literature on policy gradient methods (Zhang et al., 2019a). In practice, a saturation function can be applied to bound the reward. Besides, the condition (2) in Theorem 3 is readily satisfied by the Gaussian policy. Please refer to Doya (2000); Zhang et al. (2019a) for more details.

4.3 Parameterisation for the policy

In this subsection, the details of the parameterisation of a policy $\boldsymbol{\pi}$ is described with the following considerations:

- The μ function is expected to approach the optimal value of γ_{k+1} suggested in (11).
- The better we understand the plant, the smaller standard deviation $\sigma(e_k,\theta)$ should be.

Parameterisation for the mean: Approximating the mean as a linear function benefits the further derivation in the policy gradient methods and provides good convergence guarantees (Sutton and Barto, 2018). When no prior is available, a general way to do is to formulate the mean as a linear combination of some basis functions, including polynomials, Fourier basis, radial basis functions (RBFs), etc. In this article, the following function approximators (features in RL) are selected to approximate μ, i.e.

$$\begin{aligned}\mu(\theta_\mu(e_k)) &= \theta_\mu^T z_\mu\\ &= \theta_\mu^T[z_{\mu,1}(e_k), z_{\mu,2}(e_k), ..., z_{\mu,m}(e_k)]\end{aligned} \quad (25)$$

where $m \ge 1$ is the number of approximators; $\{z_{\mu,i}(e_k)|i = 1,2,...,m\}$ is defined as

$$z_{\mu,i}(e_k) = \exp(-\frac{1}{2a_i^2\|e_k\|^2}) \quad (26)$$

where a_i $(m \ge 1)$ are scaling parameters.

Parameterisation for the standard derivation: It is desired that σ approaches zero as μ approaches the optimal value, so that the optimal policy is maintained. Here, $e_k^T e_k$ can be used as a measure to evaluate the distance between μ and the optimal parameter. As a small error is expected to lead to a less deviation, the σ can be selected as

$$\sigma(e_k,\theta_\sigma) = \exp(-\frac{\theta_\sigma}{\|e_k\|^2}) = \exp(\theta_\sigma z_\sigma(e_k)) \quad (27)$$

where $\theta_\sigma > 0$; $z_\sigma(e_k) = -\frac{1}{\|e_k\|^2}$ is the function approximator for σ function approximation.

With the above function approximator selection, the gradient term in (24) can be calculated as follows

$$\nabla_{\theta_\mu}\ln\pi(\gamma_{k+1}|e_k,\theta_\mu) = \frac{\gamma_{k+1} - \mu(e_k,\theta_\mu)}{\sigma(e_k,\theta_\sigma)^2}z_\mu(e_k)$$

$$\nabla_{\theta_\sigma}\ln\pi(\gamma_{k+1}|e_k,\theta_\sigma) = \left(\frac{(\gamma_{k+1} - \mu(e_k,\theta_\mu))^2}{\sigma(e_k,\theta_\sigma)^2} - 1\right)z_\sigma(e_k)$$

In the next section we will present a numerical example to verify the proposed PG-POILC.

5. NUMERICAL EXAMPLE

In this section, a numerical example is provided to demonstrate the performance of the proposed algorithms using a model of three-axis gantry robot test platform (which has been used to test varying ILC algorithms).

Each axis is controlled independently, the design problem of the Z-axis is considered as an example. According to Ratcliffe (2005), the transfer function of the Z-axis is

$$G(s) = \frac{(s+473.51)(s+199.02)}{s(s+989.06)(s^2+532.44s+95777.08)}.$$

For model-based design, we assume that a nondominant pole has not been identified and the following (inaccurate) model is used

$$G_{inaccurate}(s) = \frac{(s+473.51)(s+199.02)}{s(s^2+532.44s+95777.08)}.$$

The trial length is assumed to be 0.5s and the sampling time T_s is 0.01s (with a zero-order hold). The initial condition x_0 is set to 0 and the reference is defined as

$$r(t) = 2\sin(T_s t) \qquad t \in [0, 50] \tag{28}$$

The algorithms developed in previous sections are simulated using 200 experiments (trials) with a zero initial input condition. The parameter selection is set as follows: for model-based, data-driven and PG-POILC, the weight ω is taken to be 10^{-8}. For PG-POILC, (25) and (27) are used to describe the parameterised policy's mean and standard deviation σ, with $\{a_i\}(i = 1, 2, \dots, 10)$ selected to be the first 10 values in the sequence $\{10^0, 10^{-1}, 10^1, 10^{-2}, 10^2, \dots\}$ to facilitate a proper exploration. Besides, the step size for the stochastic gradient in (24) is set to 10^{-4} at first then decreases gradually. The parameter vector θ_μ is initialised with $0.00002 \times \mathbf{1}$, where $\mathbf{1}$ is the all-ones vector, and θ_σ is initialised with $0.01 \times \mathbf{1}$. The results are shown in Figure 1.

As can be seen from Figure 1, the model-based POILC performs the best with the fastest convergence when the system model is accurate, but its performance deteriorates significantly when an inaccurate model is used; data-driven POILC (without a model) achieves the same performance (tracking accuracy) but requires twice the number of experiments required compared to the model-based design with an accurate model; and RL-based POILC converges at a rate between model-based (with an accurate model) and data-driven ILC algorithms. Both of the proposed data-driven method and the RL-based POILC can reduce the tracking error norm effectively in a model-free manner.

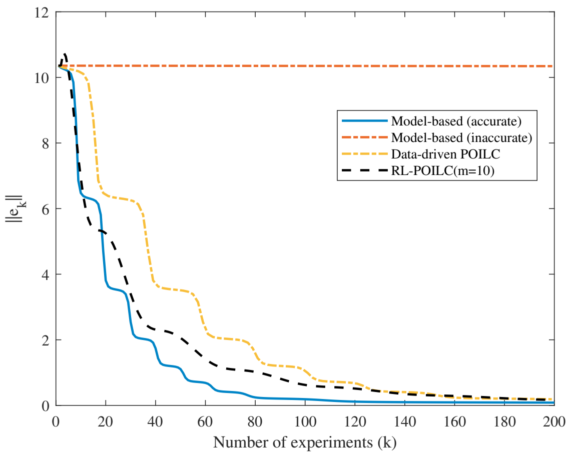

Fig. 1. Convergence of the error norm over first 200 trials

To further explore the impact of the number of function approximators on the tracking performance, different numbers of function approximators (26) are used. The parameters $\{a_i\}(i = 1, 2, \dots, m)$ for the function approximators (26) are selected to be the first m values in the sequence $\{10^0, 10^{-1}, 10^1, 10^{-2}, 10^2, \dots\}$ to ensure that only the number of function approximators differs. The tracking performance is shown in Figure 2. It can be noticed that the more function approximators used, the faster the convergence rate tends to be. Thus, the (model-free) RL-based POILC has comparable convergence performance to the model-based approach (with an accurate model).

Table 1. The number of experiments needed to achieve the tracking requirement

Method	MB (accurate)	DD	RL(m=5)
Number	135	270	999
Method	RL(m=10)	RL(m=20)	MB(inaccurate)
Number	184	114	-

To compare the convergence performance of different algorithms, the tolerated tracking error norm is set to be 0.2 and the number of experiments required to achieve this accuracy is listed in Table 1. As can be seen, the model-based POILC algorithm with an accurate model (labelled as 'MB (accurate)') achieves the accuracy target with 135 experiments; whereas the data-driven POILC algorithm (labelled as 'DD') requires twice the number of experiments. The PG-POILC (labelled as 'RL') with fewer function approximators takes much more experiments but still achieves the accuracy requirement, implying that the fewer parameters in the parameterisation are likely to result in a slower convergence to achieve the same precision. On the other hand, the PG-POILC with more function approximators takes fewer experiments than the model-based one, which suggests that scaling parameters do enhance the exploratory ability and enable a faster model-free parameter optimisation. Note that, the model-based POILC algorithm with an inaccurate model (labelled as 'MB (inaccurate)') fails to meet the desired accuracy. These findings are consistent with our previous analysis and observations. In addition, the computation effort for different algorithms are compared in Table 2. The model-based POILC takes the least time to update the input, following with the PG-POILC. Data-driven POILC contains an extra experiment on the plant (which includes one trial length) thus takes the longest time to update the input. To conclude, the data-driven POILC and PG-POILC outperform the model-based design when an accurate model of the system dynamics is not available; PG-POILC takes less computation time for an update than data-driven POILC.

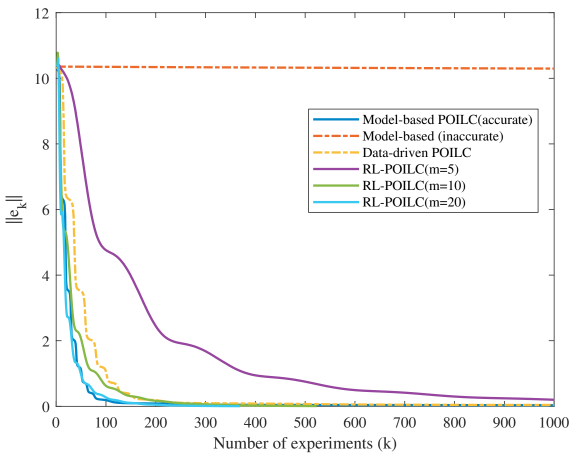

Fig. 2. Convergence of the error norm over first 1000 trials

Table 2. The computation time of each input update for different POILC algorithms

Method	MB	DD	RL (m=5)
Time	9 μs	11 μs + 0.5s	15 μs
Method	RL(m=10)	RL(m=20)	MB(inaccurate)
Time	18 μs	28 μs	9 μs

6. CONCLUSIONS

Our earlier work (Zhang et al., 2019b) showed that RL shares many similarities with ILC and therefore can be used to develop new model-free ILC algorithms. Along this line of research, this paper studies the POILC design and proposes a data-driven POILC algorithm (by conducting an extra experiment between two trials) and an RL-based POILC algorithm (using the policy gradient method) to solve the high-performance tracking problem for systems with unknown dynamics. Both of the proposed algorithms can solve the ILC design problem and achieve fast convergence of the tracking error norm to zero. Moreover, when the number of function approximators in RL-based algorithm (without a model) increases, its convergence performance approaches that of the model-based solution (with an accurate model). The simulation results confirm the effectiveness of the proposed algorithms.

While good performance has been achieved with reinforcement learning-based algorithms, there is a need to quantify the rates of convergence and tune algorithm parameters accordingly to further improve the performance of the algorithm, as well as extensions of this method to more general ILC design problems, including noisy measurements, constraint handling, varying tracking tasks, and nonlinear systems. The above constitutes part of our future research and will be reported separately.

REFERENCES

Amann, N., Owens, D.H., and Rogers, E. (1996). Iterative learning control using optimal feedback and feedforward actions. *International Journal of Control*, 65(2), 277–293.

Bertsekas, D.P. (2019). *Reinforcement learning and optimal control.* Athena Scientific.

Bolder, J., Kleinendorst, S., and Oomen, T. (2018). Data-driven multivariable ILC: Enhanced performance by eliminating L and Q filters. *International Journal of Robust and Nonlinear Control*, 28(12), 3728–3751.

Bristow, D.A., Tharayil, M., and Alleyne, A.G. (2006). A survey of iterative learning control: A learning-based method for high-performance tracking control. *IEEE Control Systems Magazine*, 26(3), 96–114.

Doya, K. (2000). Reinforcement learning in continuous time and space. *Neural Computation*, 12(1), 219–245.

Harte, T.J., Hätönen, J., and Owens, D.H. (2005). Discrete-time inverse model-based iterative learning control: stability, monotonicity and robustness. *International Journal of Control*, 78(8), 577–586.

Janssens, P., Pipeleers, G., and Swevers, J. (2012). A data-driven constrained norm-optimal iterative learning control framework for LTI systems. *IEEE Transactions on Control Systems Technology*, 21(2), 546–551.

Norrlöf, M. (2002). An adaptive iterative learning control algorithm with experiments on an industrial robot. *IEEE Transactions on Robotics and Automation*, 18(2), 245–251.

Owens, D.H., Hätönen, J.J., and Daley, S. (2009). Robust monotone gradient-based discrete-time iterative learning control. *International Journal of Robust and Nonlinear Control*, 19(6), 634–661.

Owens, D.H. (2015). *Iterative learning control: an optimization paradigm.* Springer.

Owens, D.H. and Feng, K. (2003). Parameter optimization in iterative learning control. *International Journal of Control*, 76(11), 1059–1069.

Park, J.U., Hardy, M., Kang, S.J., Barton, K., Adair, K., kishore Mukhopadhyay, D., Lee, C.Y., Strano, M.S., Alleyne, A.G., Georgiadis, J.G., et al. (2007). High-resolution electrohydrodynamic jet printing. *Nature Materials*, 6(10), 782–789.

Poot, M., Portegies, J., and Oomen, T. (2020). On the role of models in learning control: Actor-critic iterative learning control. In *IFAC 21st Triennial World Congress, Berlin, Germany.*

Ratcliffe, J.D. (2005). *Iterative learning control implemented on a multi-axis system, University of Southampton.* Ph.D. thesis, PhD Thesis.

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

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

Zhang, K., Koppel, A., Zhu, H., and Başar, T. (2019a). Convergence and iteration complexity of policy gradient method for infinite-horizon reinforcement learning. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, 7415–7422. IEEE.

Zhang, Y., Chu, B., and Shu, Z. (2019b). A preliminary study on the relationship between iterative learning control and reinforcement learning. *IFAC-PapersOnLine*, 52(29), 314–319.

Zhang, Y., Chu, B., and Shu, Z. (2022). Model-free predictive optimal iterative learning control using reinforcement learning. 2022 American Control Conference (ACC), to appear.

Appendix A. PROOF FOR THE THEOREM 2

Proof.

$$\begin{aligned} J_{k+2}(0) &= \|e_k\|^2 \geq \min_{\gamma_{k+2}} J_{k+2}(\gamma_{k+1}) \\ &= J_{k+2}(\gamma^*_{k+2}) = \|e_{k+2}\|^2 + \omega {\gamma^*_{k+2}}^2, k = 0, 2, 4, ... \end{aligned} \tag{A.1}$$

Thus, $\|e_{k+2}\|^2 \leq \|e_k\|^2$ for any even k. Thus, the monotonically convergence property in equation (16) is proven. As for the zero convergence property in equation (17),

$$G + G^T > 0 \iff \exists \epsilon > 0, \quad \text{s.t.} \quad G + G^T > \epsilon^2 I$$

and hence

$$\gamma^*_{k+1} = \frac{e_k^T G e_k}{\omega + e_k^T G^T G e_k} \geq \frac{e_k^T G e_k}{\omega} \geq \frac{\epsilon^2}{2\omega}\|e_k\|^2 \geq 0$$

Besides, applying equation (A.1) recursively gives

$$\|e_0\|^2 \geq \|e_2\|^2 + \omega {\gamma_2^*}^2 \geq ... \geq \|e_{k+2}\|^2 + \omega \sum_{i=0}^{k} {\gamma^*_{i+2}}^2$$

which suggest $\sum_{i=0}^{k} {\gamma^*_{i+2}}^2 < \infty$, thus $\lim_{k\to\infty} \gamma^*_{k+1} = 0$. Accordingly, $\lim_{k\to\infty} e_k = 0$ for any even k.