Journal of Process Control 19 (2009) 1589–1600

Contents lists available at ScienceDirect

Journal of Process Control

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

Review

Survey on iterative learning control, repetitive control, and run-to-run control

Youqing Wang [a,b], Furong Gao [c], Francis J. Doyle III [a,b,*]

[a] *Department of Chemical Engineering, University of California Santa Barbara, Santa Barbara, CA, USA*
[b] *Biomolecular Science and Engineering Program, University of California Santa Barbara, Santa Barbara, CA, USA*
[c] *Department of Chemical and Biomolecular Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong*

ARTICLE INFO

Article history:
Received 1 December 2008
Received in revised form 11 September 2009
Accepted 11 September 2009

Keywords:
Iterative learning control
Repetitive control
Run-to-run control
Repetitive process
Batch process
Run-based process

ABSTRACT

In this paper, three control methods—iterative learning control (ILC), repetitive control (RC), and run-to-run control (R2R)—are studied and compared. Some mathematical transformations allow ILC, RC, and R2R to be described in a uniform framework that highlights their similarities. These methods, which play an important role in controlling repetitive processes and run-based processes, are collectively referred to as *learning-type control* in this paper. According to the classification adopted in this paper, learning-type control has two classes—*direct form* and *indirect form*. The main ideas and designing procedures for these two patterns are introduced, separately. Approximately 400 papers related to learning-type control are categorized. Statistical analysis of the resulting data reveals some promising fields for learning-type control. Finally, a flowchart based on the unique features of the different methods is presented as a guideline for choosing an appropriate learning-type control for different problems.

Contents

* Corresponding author. Address: Department of Chemical Engineering, University of California Santa Barbara, Santa Barbara, CA, USA.
E-mail address: frank.doyle@icb.ucsb.edu (F.J. Doyle).

0959-1524/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jprocont.2009.09.006

1. Introduction

The control methods of iterative learning control (ILC), repetitive control (RC), and run-to-run control (R2R) were originally proposed in different fields, to address different problems, and were proposed by different authors; therefore, they have different formulations, different specializations, and different defining characteristics. All of them, however, use previous information to design a new control signal. That is to say, these methods can "learn" from experience to improve control performance, so ILC, RC, and R2R are categorized as *learning-type control*. In this paper, these three methods are reviewed in alphabetical order and given equal priority.

The similarities between ILC, RC, and R2R have been noted by other scholars. For example, in [1], ILC and RC were compared based on experimental results. Some similarities between ILC and R2R were mentioned in [2]. In 2007, Prof. J.H. Lee and Prof. K.S. Lee presented an overview of ILC [3], in one section of which RC and R2R were compared with ILC. However, to the best knowledge of the authors, there is no existing paper in which these three methods have been compared in detail. This article will describe ILC, RC, and R2R in a uniform mathematical framework and illustrate their essential similarities.

Many books and surveys have focused on ILC, RC, or R2R, individually. Each of them cite many references; for example, an overview paper on ILC [4] has 514 references; a survey on repetitive control [5] includes 107 references; while a survey about run-to-run control [6] cites 42 references. It is inappropriate to re-catalogue all those citations. Hence, only references indispensable to this paper are cited; those references not cited have been omitted for space concerns only. To capture the information in additional references, about 400 articles are reviewed and categorized in Section 6; however, the majority of them are not cited as references.

The preceding categorization reveals which fields have numerous reported works and which fields are more open in terms of research opportunities. Based on these observations, some promising directions will be introduced, which will be helpful for theoretical studies in learning-type control.

In general, for a process that is repetitive and/or cyclic in nature, the learning-type control method should be the first choice for control. The specific type of learning-type control should then be selected according to the characteristics of the process. Some guidelines for choosing the appropriate learning-type control method for different problems will be introduced; these will be valuable for people who are interested in using learning-type control methods, particularly for non-experts.

The objectives of this paper include: (1) describing ILC, RC, and R2R in a uniform mathematical framework and illustrating their similarities; (2) making clear their distinctions; (3) revealing some promising directions for the future; and (4) presenting guidelines for choosing the appropriate learning-type control methods.

In the following, both "continuous process" and "continuous system" will be mentioned; however, they are often taken to be the same thing by mistake; hence, their definitions will be presented first. Most processes can be divided into two classes—*batch processes* and *continuous processes*. In general, batch processes, which run intermittently, are best suited to low-volume and high-value products, while continuous processes are good at making high volume products continuously. A dynamic system is a function of time, and it likewise can be classified in two ways. If the time variables only hold discrete values, the system is called a *discrete-time system*, or *discrete system* for short. While if the time variables hold continuous values, the system is considered a *continuous-time system*, or *continuous system* for short. Hence, "continuous process" and "continuous system" are different concepts based on distinct standards. A continuous process may be discrete-time or continuous-time. Similarly, a continuous system can be achieved by a continuous process as well as a batch process.

This paper focuses mainly on discrete systems, for two reasons. First, discrete learning-type control is easier to use. Information storage is difficult in continuous systems, so in practice continuous systems are usually approximated by discrete models for the purpose of designing a learning-type controller. Second, the continuous system under repetitive control is infinite dimensional, while the discrete system under repetitive control is finite dimensional, which is easier to study. In addition, most learning-type controls have been used in linear systems, and even when the system is nonlinear, the learning-type control is designed based on a linear model of the nonlinear system in most cases. Hence, the systems considered in this paper are limited to linear discrete systems.

The rest of this paper is organized as follows: The conventional learning-type methods are introduced and compared in Section 2. Section 3 discusses how to use real-time information, which introduces one of the main differences between ILC/RC and R2R. Direct learning-type control is discussed in Section 4 and the indirect learning-type control is introduced in Section 5. Topical publications are reviewed and categorized in Section 6. A brief outlook is presented in Section 7. A guideline for choosing the appropriate learning-type control law is introduced in Section 8. Section 9 gives the conclusions.

2. Overview of ILC, RC, and R2R

2.1. Iterative learning control

In 1978, Uchiyama presented the initial explicit formulation of ILC in Japanese [7]. In 1984, Arimoto et al. first introduced this method in English [8]. These contributions are widely considered to be the origins of ILC. One motivation for the development of ILC is the industrial robot, which repeats the same task from trial to trial. Humans can learn from repeat training, and scholars have tried to find a scheme to implement such a learning ability in the automatic operation of dynamic systems. This scheme is known as iterative learning control, and has mainly focused on batch processes.

A batch process, which repetitively performs a given task over a period of time (called a batch or trial), can be described as follows:

$$\begin{cases} x(t+1,k) = Ax(t,k) + Bu(t,k) + w(t,k) \\ y(t,k) = Cx(t,k) + v(t,k) \end{cases} \tag{1}$$
$$t = 0, 1, \ldots, T-1; \quad k = 1, 2, \ldots$$

where $x \in \mathbb{R}^n$, $u \in \mathbb{R}^p$, $y \in \mathbb{R}^m$, $w \in \mathbb{R}^n$, and $v \in \mathbb{R}^m$ denotes states, inputs, outputs, disturbances or uncertainties, and measurement noise, respectively; A, B, and C are the system matrices with appropriate dimensions; t denotes different time step and T is the *duration* of each batch, and k is the batch index. Because the initial state of each batch can usually be reset in many applications, the following assumption is widely used in existing publications,

$$x(0,k) \equiv x_0 \tag{2}$$

The common control objective for batch processes is to make the outputs track the *reference trajectory* $y_r(t)$ as closely as possible. Due to (2), $y(0, k) \equiv y_0$, where y_0 is available. Hence, $y_r(0) = y_0$ is required in the earlier period of development of ILC [8]. The results of theoretical analysis show that this constraint on the reference is not necessary [9]; therefore, the easiest way to design the reference is such that $y_r(t) \equiv Yr$, where Yr is also called the *target* for the output. The initial condition for ILC is an interesting problem; some further discussion of this problem will be provided in Section 3.4.

One can define

$$e(t,k) \hat{=} y_r(t) - y(t,k) \tag{3}$$

as the *tracking error*. One of the objectives of ILC is that $\lim_{k\to\infty} e(t,k) = 0$. The simplest formulation of ILC may be:

$$u(t,k) = u(t,k-1) + K_{\mathrm{ILC}} e(t,k-1) \tag{4}$$

The inputs in the current batch are determined by the inputs of the previous batch plus the proportional contribution of tracking error in the previous batch, where K_{ILC} is the *learning gain matrix*. This type of ILC is called proportional-type ILC (P-type ILC). In most cases, P-type ILC is used as follows:

$$u(t,k) = u(t,k-1) + K_{\mathrm{ILC}} e(t+1,k-1) \tag{5}$$

For a delay-free system in Eq. (1), if $x(t, k) = x(t, k-1)$ and $u(t, k) = u(t, k-1)$, then $e(t+1, k)$ can be approximated to $e(t+1, k-1)$. Therefore, $e(t+1, k-1)$ can be considered the prediction of $e(t+1, k)$ in some sense; hence, it is reasonable using $e(t+1, k-1)$ to update $u(t, k)$, which will determine $e(t+1, k)$, as shown in (5). If the information after t in previous batch is used to design $u(t, k)$, this kind of ILC is called *phase-lead type* ILC [10] or *anticipatory-type* ILC [11].

ILC can be described in the following general formulation:

$$u(t,k) = Q_{\mathrm{ILC}}(u(t,k-1)) + r(t,k) \tag{6}$$

where $Q_{\mathrm{ILC}}(u(t, k-1))$ is called the *feedforward part* of ILC; $r(t, k)$ is called the *updating law* for ILC; $Q_{\mathrm{ILC}}(\cdot)$ is called the *Q-filter*. Specially, if $Q_{\mathrm{ILC}}(u) = Q \times u$, where Q is a scalar within (0, 1), it is also called the *forgetting factor*. A frequently used form of *Q*-filter is

$$Q_{\mathrm{ILC}}(u(t,k)) = \alpha_1 u(t+1,k) + \alpha_0 u(t,k) + \alpha_1 u(t-1,k) \tag{7}$$

where $2\alpha_1 + \alpha_0 = 1$. In this case, the *Q*-filter can be considered a weighted-average operator in a symmetrical window. In general, the *Q*-filter can improve the robustness of ILC to high frequency uncertainties but results in non-zero final tracking error. In much of the literature, $Q_{\mathrm{ILC}}(u) = u$. Most linear updating laws can be written in the following form:

$$r(t,k) = L_{\mathrm{ILC}}(e(t,k-1)) \tag{8}$$

where $L_{\mathrm{ILC}}(\cdot)$ is called the *L-filter*. For example, if $L_{\mathrm{ILC}}(e) = K_{\mathrm{ILC}} \times e$, this leads to a P-type ILC; if $L_{\mathrm{ILC}}(e(t, k)) = K_{\mathrm{ILC}} \times (e(t, k) - e(t-1, k))$, this is a D-type ILC.

2.2. Repetitive control

In 1981, the concept of repetitive control was originally developed [12,13]. The initial motivations and representative examples include the rejection of periodic disturbances in a power supply control application [12] and the tracking of periodic reference inputs in a motion control application [13]. RC is mainly used in continuous processes for tracking or rejecting periodic exogenous signals. In most cases, the period of the exogenous signal is known.

The internal model principle (IMP) proposed by Francis and Wonham [14] is the theoretical foundation of RC. According to IMP, to track or reject a certain signal without steady-state error, the signal can be regarded as the output of an autonomous generator that is inside the control system. Any periodic signal with period T can be generated by the free time-delay system shown in Fig. 1 with an appropriate initial function. A controller containing this generator is known as a repetitive controller.

Consider a linear system $Y(z^{-1}) = G(z^{-1})U(z^{-1})$, where z^{-1} is the backward shift operator; G is the transfer function; and Y and U are the z-transformations of outputs and inputs, respectively. The control objective is such that the output follows a given trajectory $R(z^{-1})$ with known period T. The simplest RC for this problem can be designed as shown in Fig. 2, where K_{RC} is termed the *repetitive control gain*. This kind of RC is also called P-type RC. The control objective of RC is to find an appropriate K_{RC} such that the tracking error $E(z^{-1})$ converges to zero as time approaches infinity. The transfer function for this RC is

$$U(z^{-1}) = \frac{K_{\mathrm{RC}} z^{-T}}{1 - z^{-T}} E(z^{-1}) \tag{9}$$

A more general formulation of RC is shown in Fig. 3, where $Q_{\mathrm{RC}}(z^{-1})$ and $L_{\mathrm{RC}}(z^{-1})$ are called the *Q-filter* and the *L-filter*, respectively. The transfer function for the general RC is

$$U(z^{-1}) = \frac{z^{-T} L_{\mathrm{RC}}(z^{-1})}{1 - z^{-T} Q_{\mathrm{RC}}(z^{-1})} E(z^{-1}) \tag{10}$$

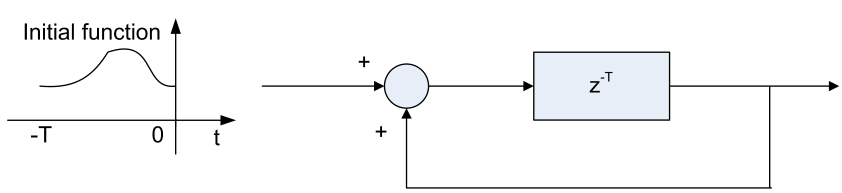

Fig. 1. Generator of periodic signal.

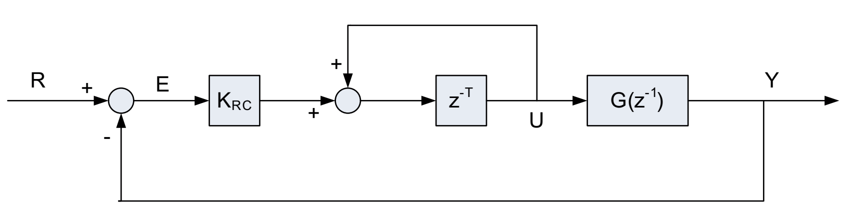

Fig. 2. Block diagram of the simplest repetitive control.

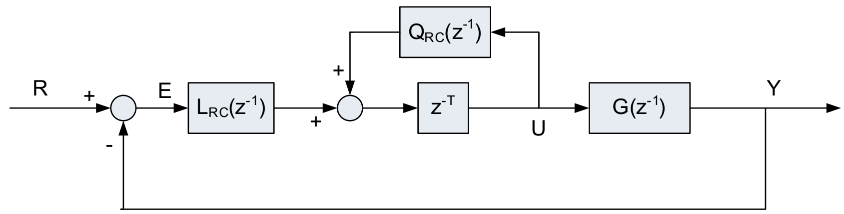

Fig. 3. General structure of repetitive control.

Table 1
Comparison of ILC/RC (ILC and RC) and R2R. ILC denotes iterative learning control; RC denotes repetitive control; R2R denotes run-to-run control.

		ILC/RC		R2R
Problem		Repetitive process		Run-based process
		ILC Batch process	RC Continuous process with periodic input; periodic continuous process	
Model		Dynamic model		Static model
		ILC State space model	RC Transfer function	
Output		Frequent measurement		Sparse measurement
Input		Varying profile		No profile OR fixed profile structure
Control structure	*t*-Direction	Closed-loop		Open-loop
	k-Direction	Closed-loop		Closed-loop

There are various schemes to design the *Q*-filter and the *L*-filter to improve the robustness of RC. A frequently used form of *Q*-filter is

$$Q_{RC}(z^{-1}) = \alpha_1 z + \alpha_0 + \alpha_1 z^{-1} \tag{11}$$

where $2\alpha_1 + \alpha_0 = 1$. Two frequently used forms for *L*-filter are $L_{RC}(z^{-1}) = K_{RC}$ and $L_{RC}(z^{-1}) = K_{RC} \times (1 - z^{-1})$.

2.3. Run-to-run control

Run-to-run (or run-by-run) control was first proposed by Sachs and his co-workers at MIT in the beginning of 1990s [15,16]. The main motivation for development of run-to-run control is the lack of *in situ* measurements for the product qualities of interest. A typical example is semiconductor manufacturing, where the goal is to control qualities, such as film thickness or electrical properties, which are difficult or impossible to measure in real-time. The considered process can be divided into several runs, which are similar to batch processes but more extensive, so it is named *run-based process* in this paper. Because only sparsely sampled outputs are available, a linear regression model is used to describe the process:

$$y(k) = Au(k) + b(k) + \varepsilon(k) \tag{12}$$

where $k = 1, 2, \ldots$ denotes run index; $y \in \mathbb{R}^m$ is the system output; $u \in \mathbb{R}^p$ is the input (recipe); A is the *slope coefficients matrix*; b is the *drift coefficients matrix*; and ε denotes disturbances. For simplicity, it is assumed that $m = p$ in this section; for other cases, please refer to pages 72–74 in [17]. The target for y is denoted as y^*. If A and b is known, then the optimal control for system (12) is

$$u(k) = A^{-1}(y^* - b(k)) \tag{13}$$

Mismatches between model and system are unavoidable in practical application, and system variations occur from run-to-run sometimes, so an iterative scheme is proposed to update the estimation of b:

$$b(k) = \lambda[y(k-1) - Au(k-1)] + (1-\lambda)b(k-1) \tag{14}$$

The preceding algorithm is called exponentially weighted moving average (EWMA) filter [17], where $0 < \lambda < 1$ is an adjustable tuning parameter for the EWMA filter. Combining (13) with (14) produces

$$\begin{aligned} b(k) &= \lambda\{y(k-1) - AA^{-1}[y^* - b(k-1)]\} + (1-\lambda)b(k-1) \\ &= b(k-1) + \lambda[y(k-1) - y^*] \end{aligned} \tag{15}$$

hence

$$u(k) = u(k-1) + \lambda A^{-1}[y^* - y(k-1)] \tag{16}$$

This algorithm is called *EWMA-type R2R* or *P-type R2R*. A more general form of R2R is as follows,

$$u(k) = \alpha u(k-1) - r(k) \tag{17}$$

where $0 < \alpha < 1$ is the *forgetting factor* and $r(k)$ is called the *updating law* for R2R.

2.4. Comparison and uniform formulation

From (6) and (10), it can be found that the control signals produced by ILC and RC are functions of time, so the inputs have profile in each batch or period. From (17), it is known that the inputs determined by R2R are constant in each run; in other words, they have no profile. In [18], a general procedure algorithm was introduced to design R2R. In the first step of this algorithm, the input profile for run k, $u_k(t)$, was parameterized as $U(t, v_k)$. Then the conventional R2R was used to update v_k. Therefore, v_k has no profile; however, the final control signal $u_k(t)$ has a profile, but the structure of the profile function is fixed. These statements are included in Table 1. To achieve frequent updates of the control signal from ILC and/or RC, there should be multiple measurements; while in R2R, sparsely sampled output measurements are enough to allow for the control update.

Therefore, the differences between ILC, RC, and R2R are reasonably distinct. Next, it will be shown that they can be described in a uniform framework. First, RC will be reformulated in time-domain.

From (10), it is found that

$$u(t) = Q_{RC}u(t-T) + L_{RC}e(t-T) \tag{18}$$

For convenience, the following operator is introduced.

$$\bar{\xi}(\bar{t}, \bar{k}) \hat{=} \xi(t) \quad \text{where } \bar{k} = \lfloor t/T \rfloor,\ \bar{t} = t - \bar{k}T,\ \xi = u, y, e \tag{19}$$

$\lfloor * \rfloor$ returns the nearest integer for $*$ towards minus infinity. Operation (19), called the *batch-operator* for short, divides the continuous sequence into several batches. By using the batch-operator, (18) can be rewritten as

$$\bar{u}(\bar{t},\bar{k}) = Q_{\mathrm{RC}}\bar{u}(\bar{t},\bar{k}-1) + L_{\mathrm{RC}}\bar{e}(\bar{t},\bar{k}-1) \tag{20}$$

From (7) and (11), it can be seen that Q_{ILC} and Q_{RC} are identical; similarly, L_{ILC} and L_{RC} are also identical. According to (6) and (8), RC has been transformed to ILC. Next, ILC will be transformed to R2R. From (1), it is found that

$$\bar{\mathbf{Y}}(k) = \bar{\bar{A}}\bar{\mathbf{U}}(k) + \bar{\mathbf{b}}(k) + \bar{\boldsymbol{\varepsilon}}(k) \tag{21}$$

where

$$\bar{\mathbf{Y}}(k) \hat{=} \begin{bmatrix} y(0,k) \\ \vdots \\ y(T-1,k) \end{bmatrix}, \quad \bar{\mathbf{U}}(k) \hat{=} \begin{bmatrix} u(0,k) \\ \vdots \\ u(T-1,k) \end{bmatrix}$$

$$\bar{\mathbf{A}} \hat{=} \begin{bmatrix} 0 & 0 & \cdots & 0 \\ CB & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ CA^{T-2}B & \cdots & CB & 0 \end{bmatrix}, \quad \bar{\mathbf{b}}(k) \hat{=} \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{T-1} \end{bmatrix} x(0,k)$$

$$\bar{\boldsymbol{\varepsilon}}(k) \hat{=} \begin{bmatrix} 0 & 0 & \cdots & 0 \\ C & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ CA^{T-2} & \cdots & C & 0 \end{bmatrix} \begin{bmatrix} w(0,k) \\ \vdots \\ w(T-1,k) \end{bmatrix} + \begin{bmatrix} v(0,k) \\ \vdots \\ v(T-1,k) \end{bmatrix}$$

Therefore, the preceding equations transformed a dynamic system (1) into a static model, which is called a *lifted system* or a *lifted model* [19]. The similarities between (21) and (12) is evident. ILC (4) can be rewritten as

$$\bar{\mathbf{U}}(k) = \bar{\mathbf{U}}(k-1) + \bar{\mathbf{K}}_{\mathrm{ILC}}\bar{\mathbf{E}}(k-1) \tag{22}$$

where

$$\bar{\mathbf{K}}_{\mathrm{ILC}} \hat{=} diag\left\{\underbrace{K_{\mathrm{ILC}},\ldots,K_{\mathrm{ILC}}}_{T}\right\}, \quad \bar{\mathbf{E}}(k-1) \hat{=} \begin{bmatrix} e(0,k-1) \\ \vdots \\ e(T-1,k-1) \end{bmatrix} \tag{23}$$

If there is a Q-filter (7), the following gain matrix should be added to multiply $\bar{\mathbf{U}}(\mathbf{k}-\mathbf{1})$ in (22).

$$\bar{Q} \hat{=} \begin{bmatrix} \alpha_0 & \alpha_1 & 0 & \cdots & 0 & 0 \\ \alpha_1 & \alpha_0 & \alpha_1 & \cdots & 0 & 0 \\ 0 & \alpha_1 & \alpha_0 & \ddots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \ddots & \alpha_0 & \alpha_1 \\ 0 & 0 & 0 & \cdots & \alpha_1 & \alpha_0 \end{bmatrix} \tag{24}$$

Similarly, the L-filter (8) can be expressed as a gain matrix. Therefore, a linear ILC can be transformed to a lifted form (22), which can be considered R2R.

Through transforming RC to ILC and transforming ILC to R2R, a uniform framework for ILC, RC, and R2R can be introduced as follows,

$$\bar{\mathbf{U}}(\mathbf{k}) = \bar{\mathbf{Q}}\bar{\mathbf{U}}(\mathbf{k}-\mathbf{1}) + \bar{\mathbf{F}}(\bar{\mathbf{Y}}(\mathbf{k}-\mathbf{1}),\mathbf{y_r}) \tag{25}$$

where $\bar{U}$ denotes the inputs in R2R or input sequence in ILC and/or RC; $\bar{F}(\cdot,\cdot)$ is a linear or nonlinear function of previous information and set-point. This united form reveals the essence of learning-type control methods—learning from previous experience to improve the new control signal. For convenience, the "batch" in batch processes, "period" in continuous processes, and "run" in run-based processes are called "cycle" for uniformity.

3. Real-time information

The measurement information in the current cycle is denoted *real-time* information. As shown in (25), real-time information is not used in the learning-type control law. To improve the robustness of learning-type control to non-repetitive variations in cycle direction, real-time information should be integrated.

3.1. Real-time ILC

Usually, real-time information can be included in one of two ways: either based on a real-time updating law or based on "plug-in type".

For the first type, ILC is chosen as (6) and the updating law has the following form,

$$r(t,k) = r_p(t,k) + r_c(t,k) \tag{26}$$

where $r_p(t,k)$ and $r_c(t,k)$ are terms based on the information in previous and current cycles, respectively. It has been discussed in Section 2.1 how to design $r_p(t,k)$, as shown in (8). All these methods can be used to design $r_c(t,k)$, except for the *anticipatory-type.*

The phrase "plug-in type" is borrowed from RC, as stated in Section 3.2. The typical form of plug-in type ILC [20] is

$$u(t,k) = u_{\mathrm{RT}}(t,k) + u_{\mathrm{ILC}}(t,k) \tag{27}$$

where $u_{\mathrm{RT}}(t,k)$ is a normal real-time feedback controller, such as PID, MPC; $u_{\mathrm{ILC}}(t,k)$ is a normal ILC.

3.2. Real-time RC

Similar to Section 3.1, all RCs with real-time information can also be divided into two classes. The simplest real-time updating law based RC is shown in Fig. 4. The transfer function of this controller is

$$U(z^{-1}) = \frac{K_{\mathrm{RC}}}{1-z^{-T}}E(z^{-1}) \tag{28}$$

Plug-in type RC is another popular way for including real-time information [21]. The general block diagram of plug-in type RC is shown in Fig. 5. The main idea is that there are two channels from tracking error to control input—one is a conventional RC, and another is a real-time feedback control.

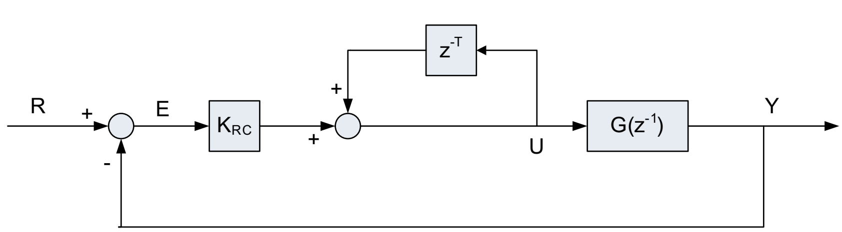

Fig. 4. The simplest real-time updating law based repetitive control.

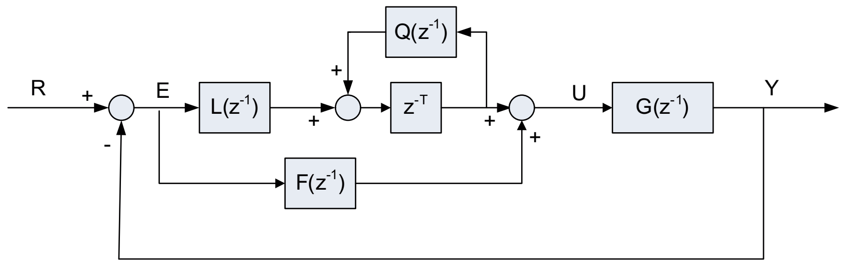

Fig. 5. Typical structure of a plug-in type repetitive control. $F(z^{-1})$ denotes the transfer function of real-time feedback control.

3.3. Real-time Information and R2R

In our opinion, real-time information cannot be used in R2R control. There are two reasons. First, it is commonly assumed that frequent measurements are not available for designing R2R; otherwise, ILC or RC would be a better choice. Second, even if frequent measurements are available, according to causality, the inputs should be determined before the measured outputs in current cycle are available. This issue will be further discussed in Section 5.

3.4. Fusion of ILC and RC

Whether the control method can accommodate real-time information is one of the main distinctions between ILC/RC and R2R. Next the essential distinctions between ILC and RC will be examined.

From Sections 2.1, 2.2 and 2.4, it can be observed that there are two main differences between ILC and RC. First, ILC and RC are typically applied in batch processes and continuous processes, respectively. Second, ILC and RC were developed in the time-domain and the frequency-domain, respectively. Considering the first distinction, however, ILC has been successively applied to continuous processes with periodic inputs [22]. This kind of ILC is also called no-reset ILC [23]. The second distinction has been mentioned implicitly in the literature, for example, in page 1163 of [5], it was stated that "most analysis and design of repetitive control are performed in the frequency-domain, which makes the nonlinear study more difficult than ILC developed in the state space"; in page 546 of [24], it was presented that "RC is primarily a frequency-domain technique". However, there is no reported work stating this distinction explicitly, which is one of the motivations for this paper.

Because ILC and RC were designed by using different tools, they have different formulations, but their essential features are nearly equivalent, and the only criteria to distinguish ILC and RC are time-domain or frequency-domain formulation. In this section, ILC and RC are considered to be the same algorithm, labeled ILC/RC.

Now the 2-dimensional (2D) model frequently used in designing ILC will be introduced [25]. Define

$$\begin{aligned}&\Delta^T\xi(t,k)\hat{=}\xi(t,k)-\xi(t-1,k),\\&\Delta^K\xi(t,k)\hat{=}\xi(t,k)-\xi(t,k-1);\quad \xi=x,u,y\end{aligned}\tag{29}$$

Given $Q_{ILC}=1$ in (6), it is obtained that

$$\Delta^K u(t,k)=r(t,k)\tag{30}$$

From (1) and (30), it can be found that

$$\Delta^K x(t+1,k)=A\Delta^K x(t,k)+Br(t,k)+\Delta^K w(t,k)\tag{31}$$

From (1) and (3), it can be found that

$$\begin{aligned}e(t+1,k)&=e(t+1,k-1)-C\Delta^K x(t+1,k)-\Delta^K v(t+1,k)\\&=e(t+1,k-1)-CA\Delta^K x(t,k)-CBr(t,k)\\&\quad -C\Delta^K w(t,k)-\Delta^K v(t+1,k)\end{aligned}\tag{32}$$

Combining (31) and (32), the following can be obtained,

$$\begin{aligned}\begin{bmatrix}\Delta^K x(t+1,k)\\e(t+1,k)\end{bmatrix}&=\begin{bmatrix}A&0\\-CA&0\end{bmatrix}\begin{bmatrix}\Delta^K x(t,k)\\e(t,k)\end{bmatrix}\\&\quad+\begin{bmatrix}0&0\\0&I\end{bmatrix}\begin{bmatrix}\Delta^K x(t+1,k-1)\\e(t+1,k-1)\end{bmatrix}+\begin{bmatrix}B\\-CB\end{bmatrix}r(t,k)\\&\quad+\begin{bmatrix}I\\-C\end{bmatrix}\Delta^K w(t,k)+\begin{bmatrix}0\\-I\end{bmatrix}\Delta^K v(t+1,k)\end{aligned}\tag{33}$$

or

$$\begin{aligned}\begin{bmatrix}\Delta^K x(t+1,k)\\e(t+1,k)\end{bmatrix}&=\begin{bmatrix}A&0\\-CA&I\end{bmatrix}\begin{bmatrix}\Delta^K x(t,k)\\e(t+1,k-1)\end{bmatrix}+\begin{bmatrix}B\\-CB\end{bmatrix}r(t,k)\\&\quad+\begin{bmatrix}I\\-C\end{bmatrix}\Delta^K w(t,k)+\begin{bmatrix}0\\-I\end{bmatrix}\Delta^K v(t+1,k)\end{aligned}\tag{34}$$

Models (33) and (34) are the 2D *Fornasini–Marchesini model* and the 2D *Roesser model*, respectively [26]. Based on these transformations, designing ILC/RC for batch process (1) is equivalent to designing a feedback control for a 2D system (33) or (34). As pointed out in [27], the convergence properties of ILC can be studied in two directions: the time direction (t-direction) and the batch direction (k-direction). Using the lifted model introduced in Section 2.4, only the stability of the closed-loop system in k-direction can be studied, so the controlled system is generally assumed to be open-loop stable. This assumption can be removed in the 2D model framework. Hence, the 2D model framework provides more freedom than the lifted model.

Based on the 2D model, the comparison of ILC, RC, and R2R is given in Table 1. In row 2, repetitive processes include repetitive batch processes, continuous processes with periodic exogenous inputs, and periodic continuous processes [28]. In row 3, dynamic model include state space model and transfer function. The formulations of these two models were introduced in Sections 2.1 and 2.2. These kinds of processes have good repetitive nature, such as even cycle durations, while run-based processes have more flexibility, for example, the duration might have huge variations from cycle to cycle, as explained in Sections 5 and 8. In row 6, the control structures under learning-type control laws are compared. Because the real-time information is not used in R2R, the system under R2R is open-loop in the t-direction.

There are two application modes to use the learning-type control. First, the learning-type control method is used to determine the control signal directly, and this kind of learning-type control is called direct learning-type control. Second, there is a local feedback controller in each cycle and the learning-type control is used to update the parameter settings of the local controller, so this kind is called indirect learning-type control. The methods that can be used for designing direct learning-type control and indirect learning-type control will be discussed in Section 4 and in Section 5, respectively.

4. Direct learning-type control

In principle, all feedback control methods can be used in plug-in type ILC/RC, so plug-in type ILC/RC will not be included in this section. There are a number of papers on higher-order ILC/RC [29,30], but they are outside the scope of this paper.

4.1. Direct ILC

The procedure to design ILC can be divided into three steps: choosing the updating law's structure, designing the Q-filter, and including some estimation schemes.

The structure of updating law is the key point of designing ILC. PID-type [31,32] should be the most common structures of the updating law. Because the stability analysis is convenient, optimal control scheme [33] is often used to design the updating law. Due to its superior abilities to deal with constraints, nonlinearities, and multi-variants, model predictive control (MPC) has been used widely to design the updating law [34–36]. From (42) in [35], it is easy to find that MPC-based ILC is an anticipatory-type ILC. In [37], convergence property of constrained MPC-based ILC was investigated. In addition, some nonlinear algorithms, such as neural network [38] and fuzzy logic [39], have been used to design the updating law. Dynamic output feedback control might be a promising candidate for ILC's updating law [40]. In our opinion, ILC is such a flexible framework that most existing feedback control methods can be used to design the updating law.

There are mature methods for design of the Q-filter. In many cases, it is chosen as $Q = 1$. Another common way is designing the Q-filter as (7).

After the structure of ILC has been determined, there might exist some unknown terms, such as states and parameters, so some schemes are needed to estimate these unknown terms. For instance, the system states included in ILC can be estimated by Kalman filter [41] or state observers [42]; some design parameters of ILC, such as the learning gain, can be searched by a recursive algorithm [43]; some system parameters can be identified by an online system identification scheme [44].

After the ILC design is completed, the stability of the closed-loop system under ILC should be analyzed. The Lyapunov theorem and its deductions [45] are basic methods for this problem. The lifted model introduced in Section 2.4 and the 2D model introduced in Section 3.4 are good frameworks for stability analysis of ILC.

4.2. Direct RC

When developed in the frequency-domain, all RCs are linear control algorithms (this statement will be further discussed in Section 6). Similarly to Section 4.1, three steps are taken to design an RC: designing the L-filter, designing the Q-filter, including an estimation scheme.

Designing the L-filter for RC is equivalent to designing a linear updating law for ILC. PID-type methods should be the main schemes for this issue.

The Q-filter design methods for RC are similar to that for ILC. The L-filter and Q-filter were first proposed in RC field and then transplanted to ILC field; therefore, the design procedures of these filters are similar for ILC and RC.

Due to the limitations of frequency-domain framework, estimation schemes are seldom used in RC. In general, the transfer function model is used for designing RC, so there is no system state to estimate. In some special cases, schemes may be needed to estimate other parameters. In [46], a recursive least-squares parameter identification algorithm was used to update the L-filter of RC, and the integrated control law had been experimentally applied to pulse-width modulated inverter.

The small gain theorem [47] is a popular tool for analysis of the stability of an RC [48]. After transforming the closed-loop system to linear fractional transformation (LFT) form, μ analysis and synthesis method (structured singular value method) can be used to analyze the stability [49].

4.3. Direct R2R

There are two ways to design R2R. In the first way, choosing the control as (13), schemes are designed to estimate b or A. EWMA introduced in Section 2.3 is the main method to update the estimation of b. Predictor–corrector control (PCC) algorithm, introduced on page 69 of [17], can be considered an expansion of the EWMA that adds an explicit model for drift. In [50], a recursive least-squares (RLS) algorithm was used to estimate b. In few cases, b is fixed and A is estimated from cycle to cycle, as introduced in [51]. It needs be pointed out that multiplicative parameters are much more difficult to estimate than additive ones.

Another way to design R2R is by first choosing R2R as (17) and then designing an updating law and forgetting factor. Due to the simplicity of R2R, there is little freedom for updating law design. P-type updating law is the main form [18,52].

The stability analysis for R2R is much easier than that for the previous cases. Lyapunov's direct method is the main method for this issue [53].

5. Indirect learning-type control

In the case of indirect learning-type control, there are two loops in the closed-loop system as shown in Fig. 6: the inner loop is the local controller and while the outer loop is the learning-type control algorithm. In this case, the learning-type control acts as a supervision or optimization module for the closed-loop system under the local controller. In fact, R2R was originally proposed in this form. In [16], an indirect R2R was proposed for plasma etching, where the local controller was designed by using the algorithms proposed in [54] and R2R was used to update the target values for the *in situ* measurements. Generally, the indirect learning-type control can be used to update the reference or set-point, control

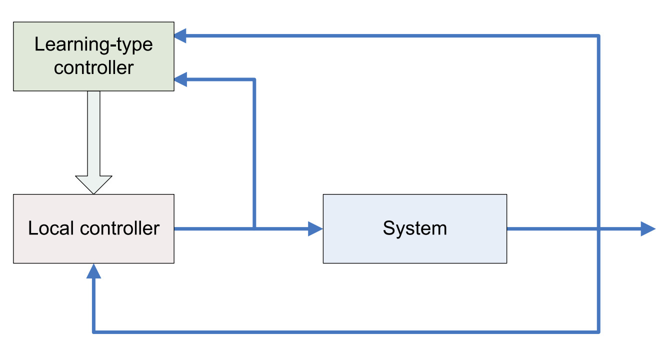

Fig. 6. Structure of indirect learning-type control. The narrow arrow line denotes the measurement information; the wide arrow line denotes the management decision.

gain, working duration and other parameters. An indirect learning-type control is made up of a common learning-type control and a local controller. Section 4 focuses on how to design the common learning-type control, and there exist many methods that can be used to design the local controller. In principle, any real-time feedback control law can be chosen as the local control. The key point for designing indirect learning-type control is to choose the appropriate variable related to the local control that could be optimized by learning-type method. Therefore, the focus of this section is on what kind of local control is used and what variables are updated.

5.1. Indirect ILC

To combine adaptive control with ILC, many researchers prefer to formulate the combination as an indirect ILC form [55,56]. The indirect formulation was utilized in this case mainly for mathematical convenience, and the supervision function of ILC is not evident. In 1995, Bone [57] used ILC to update the disturbance estimation in the prediction model for GPC, and this combination scheme had been applied to a PUMA-762 robot. In [58], a neural-network-based controller was proposed for trajectory imitation of robot manipulators and ILC was used to update the weight of neural network. In [11,59], the local controls were designed by using PID and the references for PID control were optimized by ILC. The statistical results for the literature analysis in Section 6 will show that less than 10% of ILCs are of indirect form.

5.2. Indirect RC

In fact, as early as 1988, an indirect RC was proposed in [48]. A two-degree-of-freedom control worked as the local control, and a plug-in type RC was used to update the reference trajectory for the local control. RC was also used to update the reference trajectory for the local tracking controller in [60], and this algorithm was applied to pulse-width-modulated inverter. In [61], the gain matrices of a dynamic output feedback controller were scheduled by RC. This algorithm has been verified in rotational velocity regulation in a laser printer. As shown in Section 6, indirect RC comprises less than 10% of all RC.

5.3. Indirect R2R

R2R is frequently used to schedule the working duration (how long each run will last), e.g., for shallow trench isolation etch processes [50] and chemical vapor deposition [62]. In most of these papers, the local controller is not mentioned, because it is not the emphasis or there is no local control. Even if no local control is used, R2R for working duration is also considered to be indirect due to the special property of time, and the local control is open-loop in this case. Updating the set-point for the local control is another common application of indirect R2R. In [63], R2R was used to optimize the set-point for PID. Nonlinear real-time control and R2R were combined to control etch depth and spatial uniformity in reactive ion etching [64]. Fig. 3 in [65] shows a typical block diagram of indirect R2R.

Most R2Rs are utilized in the indirect form, which is the essence of R2R. In this situation, the real-time information can be used by the local controller, but the information used to update R2R must come from the previous cycle; therefore, the statement about real-time information and R2R mentioned in Section 3.3 is still correct.

Table 2
Publications of learning-type control from Web of Science and IEEE Xplore. ILC denotes iterative learning control; RC denotes repetitive control; R2R denotes run-to-run control; Full Text denotes Full Text & All Fields; Title for IEEE Xplore denotes Document Title.

	Web of Science		IEEE Xplore	
	Topic	Title	Full Text	Title
ILC	440	254	1641	400
RC	245	106	1345	188
R2R	79	32	283	36

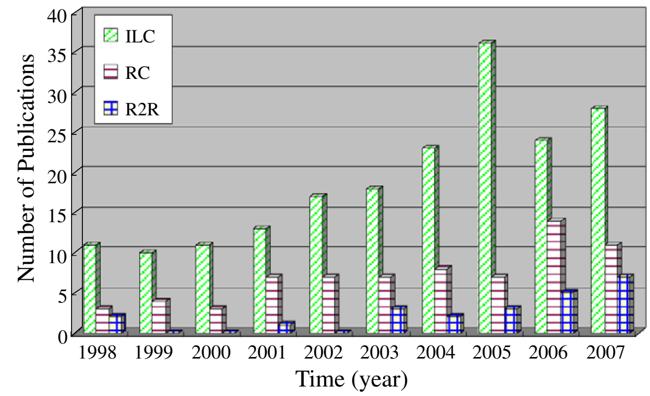

Fig. 7. Publication numbers of literature with the titles "ILC", "RC", and "R2R" in the Web of Science database in the last 10 years. ILC denotes iterative learning control; RC denotes repetitive control; R2R denotes run-to-run control.

6. Literature overview

All results presented in this section were found via literature searches in the Web of Science[1] and the IEEE Xplore[2] databases on August 8, 2008. Two searching fields, Title and Topic, were chosen in Web of Science, while Document Title and Full Text & All Fields were chosen in IEEE Xplore. The searching phrases were "iterative learning control", "repetitive control", and "run-to-run control", respectively. Please note that the quotation mark was included to make sure that unrelated papers were not involved. Because the exact names were used in the literature search, it is possible that many important learning-type control publications were missed; however, the statistics in this section still can provide some essential facts. Table 2 shows the numbers of publications about three searching phrases in four different searching fields. The numbers of publications with title "ILC", "RC" and "R2R" in Web of Science between 1998 and 2007 are shown in Fig. 7, and while the numbers of publications in IEEE Xplore from 1998 to 2007 are shown in Fig. 8. These figures demonstrate that publication numbers of learning-type control grew steadily in the recent 10 years.

There are so many publications that it is not efficient to systematically review all of them. Hence in this section the literature review is restricted to SCI papers (from Web of Science) with exact title "iterative learning control", "repetitive control", and "run-to-run control". Through the electronic journal database system in the University of California Santa Barbara (UCSB), 207, 73, and 28 electronic papers have been found for ILC, RC, and R2R, respectively. The first author reviewed all these papers to separate them into direct or indirect learning-type control, simulation or experiment, and different controlled processes. The categorizations given in this section are based on the author's subjective decision.

Among 207 articles on ILC, six of them are comments or replies to comments; three of them are survey papers; one paper is book

[1] http://apps.isiknowledge.com/WOS_GeneralSearch_input.do?product=WOS&search_mode=GeneralSearch&SID=4CjPCbEj2mMm3@iJah4&preferencesSaved=.
[2] http://ieeexplore.ieee.org/search/advsearch.jsp.

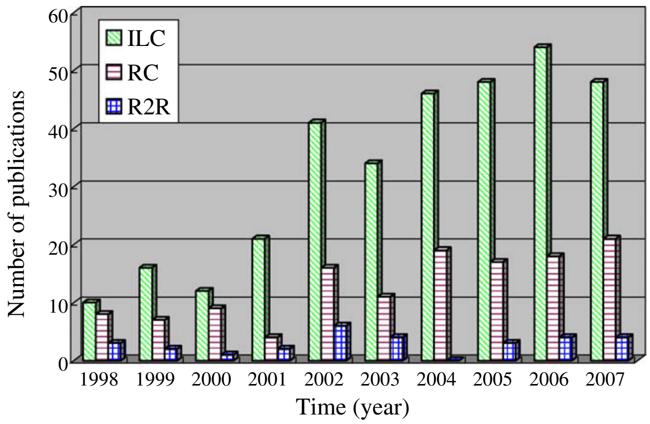

Fig. 8. Publication numbers of literature with the titles "ILC", "RC", and "R2R" in the IEEE Xplore database in the last 10 years. ILC denotes iterative learning control; RC denotes repetitive control; R2R denotes run-to-run control

Table 3
Category of learning-type control: direct and/or indirect form. ILC denotes iterative learning control; RC denotes repetitive control; R2R denotes run-to-run control.

	Direct	Indirect
ILC	177	16
RC	58	4
R2R	12	14

chapter; one paper is an editorial letter; three papers are not about ILC. Hence, only 193 publications are included in the following statistical results. Among 73 publications on RC, five articles are comments or replies to comments; two of them are survey papers; four papers are unrelated with RC. Therefore, 62 articles are studied. Concerning R2R, two papers are not in the subject area, so only 26 publications are included in the statistic results.

The categorization of learning-type control into the direct or indirect subcategories is presented in Table 3. In most cases, the distinctions between direct and indirect learning-type controls are clear. But in some cases, it is not easy to distinguish them. For example, plug-in type ILC can be viewed in two different ways: either as ILC combined with feedback control or as feedback control combined with feedforward control. In the first viewpoint, the control is a direct ILC; while in the second viewpoint, ILC is used to update the feedforward part of the local control, so it is an indirect ILC. An important criterion is that if the control input or part of the control input is determined by learning-type control directly, then the control algorithm is considered to be in the direct form.

Another puzzling case arises when there is a transformation between the learning-type control signal and the real control input. The criterion for this case is whether the transformation is related to measured output information. If the transformation is unrelated to the outputs, it cannot be considered feedback control, so the control algorithm is of the direct form, e.g. [66]. Otherwise, the algorithm is of the indirect form. In some cases [18], the control input was parameterized as $U(t, v_k)$ and R2R was used to design vector v_k. Because $U(t, v_k)$ is unrelated to the output, this kind of R2R is of the direct form. Therefore, not all direct learning-type controls design the control input directly, and there might exist a static transformation between learning-type control and actual input signal.

If R2R is used to determine the working duration, then it is categorized as indirect R2R regardless of whether local control exists. This is because time is a special physical variable whose profile cannot be designed, so the working duration cannot be considered the magnitude variable.

Table 4
Categorization of learning-type control based on application processes. ILC denotes iterative learning control; RC denotes repetitive control; R2R denotes run-to-run control.

	ILC	RC	R2R
Robotics	21	1	0
Rotary system	8	7	0
Other mechanical system	18	13	1
Semiconductor process	4	0	12
Power system	1	8	0
Chemical process	3	0	0
Optical disk system	1	6	0
Biomedical system	0	0	1
Miscellaneous	9	5	0
Total	65	40	14

Among 193 ILC-related papers, 62 RC-related papers, and 26 R2R-related papers, 65, 40, and 14 of them, respectively, provide experimental results; other papers present only simulation results or even no validation results. The 65 ILC-related papers, 40 RC-related papers, and 14 R2R-related papers that present experimental results were categorized based on their application processes. Borrowing ideas from [67,68], the following categories were chosen: "robot"; "rotary system", which includes motors and other rotating machineries; "other mechanical system", which covers non-robotic/non-motor actuators, servo systems, mechanical valve and so on; "semiconductor system"; "power system"; "chemical process"; "biomedical system"; and "miscellaneous". The statistic results on categorization of learning-type controls based on their application processes are provided in Table 4.

Among 193 ILC-related papers, nine papers proposed ILC in the frequency-domain. Strictly speaking, these algorithms should be categorized as RC, but the original classifications were maintained. Similarly, four of 62 RC-related papers presented controllers in the time-domain, so these algorithms should be denoted ILC. Correspondingly, so-called nonlinear RC could be proposed. In addition, seven of ILC-related papers and six of RC-related papers designed and analyzed control algorithms in both time and frequency-domains. It is very difficult to distinguish between ILC and RC in these situations.

7. Promising fields and outlook

As demonstrated in Figs. 7 and 8, learning-type control has been receiving increasing attention from researchers and practicing engineers. Based on the literature overview results in Section 6, some promising fields could be revealed, which will be helpful for theoretical studies.

From Table 3, most R2Rs act in the indirect form, but there are few works on indirect ILC or RC. Compared to direct form, indirect learning-type control has some advantages. First, the existing process structure need not change; only an outer loop module is added to update some parameters of the existing control. Second, in some cases, indirect learning-type control [69] has better robustness than the direct form [70]; this is because direct learning-type control must have a feedforward term, which is sensitive to variations in cycle direction, but a feedforward term is not necessary for the local controller of the indirect algorithm. In addition, the idea of the indirect method is very advanced: stability and robustness are not the only requirements for control design, and an optimization scheme should be utilized to improve the performance. In fact, indirect R2R has been successively applied in practice; however, the reported results on indirect ILC or RC are scarce.

Due to the greater flexibility, indirect ILC/RC could achieve more optimization tasks in many cases than indirect R2R; hence, it is a promising direction in the future.

From Table 4, it is obvious that learning-type control has been applied frequently in industrial processes; however, learning-type control's potential beyond industrial processes has not been explored, e.g., there is only one result on a biomedical system [52], where run-to-run control was used to adjust prandial insulin dosing for type 1 diabetes. Taking into consideration all of the papers involved in Table 3, there are still two other papers [18,71] about insulin infusion based on run-to-run control; however, there is no article on ILC or RC for biomedical systems among the 193 ILC-related papers and 62 RC-related papers. Of course, due to the small statistical sample, this does not mean that there is not existing work on ILC or RC for biomedical systems—in fact, some related work, such as [72], have been found—however, it illustrates that the potential of implementing learning-type control to biomedical systems remains largely unfulfilled. Biomedical engineering is an exciting research field wherein control technologies can play a great role [73]. Due to the diurnal cycle, the human body and other organisms exhibit repetitive natures from day to day; therefore, biomedical systems should be logical application areas for learning-type control. Many studies indicate that the specialties of learning-type control can be exploited in these fields, such as glucose control, circadian rhythm regulation, and cell cycle control. Recently, ILC was successively applied in glucose control [69,70]. The authors believe that learning-type control for biomedical systems will have dramatic progress in the near future.

Driven by manufacturing business, the last 10 years witnessed a resurgent interest in batch processing technologies. Batch processes are the preferred manufacturing choice for low-volume and high-value products such as specialty chemicals, pharmaceuticals, consumer products, and bio-products. In general, batch processes have some degree of repetitive nature; hence, designing learning-type control for batch processes is well motivated. ILC has been successfully used to control batch processes for a long time period. On the other hand, since the class of run-based processes includes batch processes, it is logical to use R2R in batch processes. The only issue remaining is RC for batch processes. In our opinion, there are two difficulties that must be addressed: (1) frequency-domain description for batch processes, and (2) storing the previous information. To the best knowledge of the authors, there is no specialized formulation for batch processes in the frequency-domain. Therefore, the authors suggest that a batch process be considered as a continuous process, and design RC for the continuous process. As shown in Fig. 2, a time-delay module can be used to store the previous information; however, this is infeasible for batch processes, because the interval duration between two batches is varying. Hence, a memory operator should be used to replace the time-delay module.

Due to their finite duration, most batch processes show nonlinear behavior. In fact, most chemical processes exhibit nonlinearity. Therefore, designing the learning-type control for nonlinear systems is a very interesting and important problem. If indirect learning-type control is implemented, any nonlinear control method can be used to design the local controller for the nonlinear system. This is a straightforward issue for learning-type control. Therefore, only direct learning-type control is discussed in this regard. Due to the limitations of the frequency-domain framework, a linear model is required to design RC for a nonlinear system. Since only a static model is used to design R2R, nonlinear issues are not considered commonly in R2R-related papers. As pointed out in Section 3.3, R2R is an open-loop method. Hence, if a R2R is designed for a nonlinear system, the nonlinear system should be open-loop stable. Designing R2R based on nonlinear static models might be a promising direction [74].

ILC is a good framework to deal with system nonlinearity. In general, there are several ways to design nonlinear ILC. First, the updating law of ILC could be designed by using nonlinear control methods, e.g. fuzzy logic [39], neural network [38], and sliding mode control [75]. Second, the structure of the updating law is linear but with nonlinear gains [76,77]. Third, some nonlinear projections, e.g. deadzone [78] and saturation function [79], can be used to improve the robustness of ILC. In addition, by designing different control laws for odd and even batches, a special ILC scheme was proposed in [80]. Although there exist isolated results for nonlinear ILC, all of them were designed for a particular class of nonlinear systems, so general strategies for nonlinear ILC is still an open problem.

8. Guidelines for choosing learning-type control

This section will introduce some guidelines to aid engineers in choosing appropriate learning-type controllers for different problems.

If there are existing controllers in the considered processes and it is impossible or difficult to change the existing process structure, indirect learning-type control will be the first choice. As stated in Section 6, there are few reported works on indirect methods, so direct learning-type control will be a better choice for other cases. Next, the strong points of direct ILC, RC, and R2R will be discussed respectively.

If there is no frequent measurement available, R2R will be the only choice. On the other hand, if the repetitive nature of the process is not very good, R2R will work better than ILC/RC. For example, if the cycle durations have significant variation, it is difficult to design a good ILC or RC controller, because all existing work on ILC/RC have an implied assumption that the cycle durations are the same or at least similar. ILC/RC is better at controlling processes with frequent measurements and reliable repetitive nature. As previously stated, ILC denotes time-domain-based ILC/RC and RC denotes frequency-domain-based ILC/RC in this section. Hence, in order to make clear the differences between ILC and RC, the differences between time-domain and frequency-domain methods must be delineated. It is clear that the time-domain method is the only choice for nonlinear systems. It is widely accepted that for multi-input multi-output (MIMO) systems, the time-domain method is more convenient than the frequency-domain method. However, the comparison between time-domain and frequency-domain methods is debatable in the case of single-input single-output (SISO) linear systems. In our opinion, one reason for the frequency-domain method widely used in practice is that it displays some advantages in working for SISO linear systems. Therefore, RC is recommended for SISO linear systems in this paper, even though it might be unfair for ILC.

These statements mentioned previously are summarized in Table 5. The distinctions of indirect ILC, RC, and R2R are similar

Table 5
Extensions of different learning-type control methods. ILC denotes iterative learning control; RC denotes repetitive control; R2R denotes run-to-run control; SISO denotes single-input single-output; MIMO denotes multi-input multi-output.

Learning-type methods	Problems being expert	
Indirect learning-type control	Local controller exists AND impossible or difficult to change the existing structure	
Direct R2R	No frequent measurement OR bad repetitive nature	
Direct RC	Frequent measurements and good repetitive nature	SISO linear systems
Direct ILC		All systems particularly for MIMO and nonlinear ones

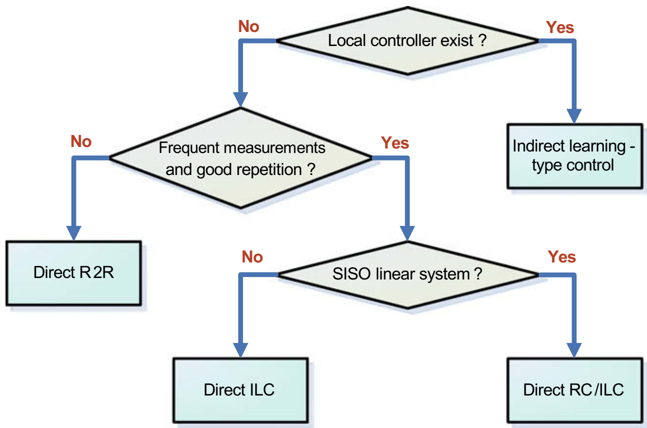

Fig. 9. The flowchart for choosing appropriate learning-type methods.

to that of direct ILC, RC and R2R. On the basis of these conclusions, a guideline for choosing appropriate learning-type control methods for different problems is presented in Fig. 9.

9. Conclusions

In this paper, the control methods ILC, RC, and R2R were compared systematically. The distinction between ILC/RC and R2R is clear. The output of R2R is constant or at least its structure is constant, while both the output of ILC/RC and its structure could be varying. The distinctions between ILC and RC are less easily defined. In our opinion, this is an issue left over from history: ILC was originally proposed in time-domain, while RC was presented in frequency-domain.

By transforming these methods into a uniform mathematical formulation, the similarities and features of learning-type control methods become much clearer. Essentially, ILC and RC are the same—updating the control signal in both time and cycle directions based on the historical data. Furthermore, designing controllers by using both time-domain and frequency-domain methods becomes more common, so it is difficult to distinct ILC and RC in some cases. Hence, it is cautiously suggested that ILC and RC could be considered the same thing in the future.

According to the application mode, learning-type control was divided into two classes—direct form and indirect form. Based on the statistical results in Section 6, indirect learning-type control is an open problem with a bright future. In order to explain the main ideas as briefly as possible, some interesting, but atypical, issues, such as higher-order, are not discussed in this paper.

Acknowledgements

This work is supported by Grant # 22-2006-1115 and 22-2007-1801 from the Juvenile Diabetes Research Foundation (JDRF) and the Institute for Collaborative Biotechnologies under Grant W911NF-07-1-0279 from the U.S. Army Research Office.

References

[1] R.W. Longman, Iterative learning control and repetitive control for engineering practice, Int. J. Control 73 (2000) 930–954.
[2] A.K. Rzepniewski, D.E. Hardt, Development of general multivariable run-by-run control methods with application to a sheet metal forming process, Int. J. Mach. Tools Manuf. 48 (2008) 599–608.
[3] J.H. Lee, K.S. Lee, Iterative learning control applied to batch processes: an overview, Control Eng. Pract. 15 (2007) 1306–1318.
[4] H.-S. Ahn, Y. Chen, K.L. Moore, Iterative learning control: brief survey and categorization, IEEE Trans. Syst. Man Cybern. Part C: Appl. Rev. 37 (2007) 1099–1121.
[5] C. Li, D. Zhang, X. Zhuang, A survey of repetitive control, in: Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, Sendai, Japan, 2004, pp. 1160–1166.
[6] E.D. Castillo, Run-to-run process control: literature review and extensions, J. Qual. Technol. 29 (1997) 184–196.
[7] M. Uchiyama, Formulation of high-speed motion pattern of a mechanical arm by trial (in Japanese), Trans. SICE (Soc. Instrum. Control Eng.) 14 (1978) 706–712.
[8] S. Arimoto, S. Kawamura, F. Miyazaki, Bettering operation of dynamic systems by learning: a new control theory for servomechanism or mechatronic system, in: Proceedings of 23rd Conference on Decision and Control, Las Vegas, Nevada, 1984, pp. 1064–1069.
[9] J. Shi, F. Gao, T.-J. Wu, Integrated design and structure analysis of robust iterative learning control system based on a two-dimensional model, Ind. Eng. Chem. Res. 44 (2005) 8095–8105.
[10] C.T. Freeman, P.L. Lewin, E. Rogers, Experimental evaluation of iterative learning control algorithms for non-minimum phase plants, Int. J. Control 8 (2005) 826–846.
[11] J.H. Wu, H. Ding, Reference adjustment for a high-acceleration and high-precision platform via A-type of iterative learning control, Proc. IMechE, Part I: J. Syst. Control Eng. 221 (2007) 781–789.
[12] T. Inoue, S. Iwai, M. Nakano, High accuracy control of a proton synchrotron magnet power supply, in: Proc. 8th IFAC World Congress, 1981, Part 2, pp. 3137–3142.
[13] T. Inoue, M. Nakano, S. Iwai, High accuracy control of servomechanism for repeated contouring, in: Proceedings of the 10th Annual Symposium on Incremental Motion Control Systems and Devices, 1981, pp. 258–292.
[14] B.A. Francis, W.M. Wonham, Internal model principle for linear-multivariable regulators, Appl. Math. Optim. 2 (1975) 170–194.
[15] E. Sachs, R.-S. Guo, S. Ha, A. Hu, On-line process optimization and control using the sequential design of experiments, in: 1990 Symposium on VLSI Technology, Honolulu, HI, 1990, pp. 99–100.
[16] E. Sachs, R.-S. Guo, S. Ha, A. Hu, Process control system for VLSI fabrication, IEEE Trans. Semicond. Manuf. 4 (1991) 134–144.
[17] J. Moyne, E. del Castillo, A.M. Hurwitz, Run-to-run Control in Semiconductor Manufacturing, CRC Press LLC, 2001.
[18] C. Owens, H. Zisser, L. Jovanovič, B. Srinivasan, D. Bonvin, F.J. Doyle, Run-to-run control of blood glucose concentrations for people with type 1 diabetes mellitus, IEEE Trans. Biomed. Eng. 53 (2006) 996–1005.
[19] W.B.J. Hakvoort, R.G.K.M. Aarts, J. van Dijk, J.B. Jonker, Lifted system iterative learning control applied to an industrial robot, Control Eng. Pract. 16 (2008) 377–391.
[20] T.-Y. Kuc, J.S. Lee, K. Nam, An iterative learning control theory for a class of nonlinear dynamic systems, Automatica 28 (1992) 1215–1221.
[21] W.-S. Yao, M.-C. Tsai, Analysis and estimation of tracking errors of plug-in type repetitive control systems, IEEE Trans. Automat. Control 50 (2005) 1190–1195.
[22] D.H. Owens, K. Feng, Parameter optimization in iterative learning control, Int. J. Control 76 (2003) 1059–1069.
[23] L.G. Sison, E.K.P. Chong, No-reset iterative learning control, in: Proceedings of the 35th Conference on Decision and Control, Kobe, Japan, 1996, pp. 3062–3063.
[24] M. Gupta, J.H. Lee, Period-robust repetitive model predictive control, J. Process Control 16 (2006) 545–555.
[25] Y.Q. Wang, J. Shi, D.H. Zhou, F.R. Gao, Iterative learning fault-tolerant control for batch processes, Ind. Eng. Chem. Res. 45 (2006) 9050–9060.
[26] T. Kaczorek, Two-dimensional Linear System, Springer, Berlin, 1985.
[27] J. Shi, F. Gao, T.-J. Wu, Robust iterative learning control design for batch processes with uncertain perturbations and initialization, AIChE J. 52 (2006) 2171–2187.
[28] S.-R. Oh, Z. Bien, I.H. Suh, An iterative learning control method with application for the robot manipulator, IEEE J. Robot. Automat. 4 (1988) 508–514.
[29] D.H. Owens, J. Hätönen, Iterative learning control – an optimization paradigm, Annu. Rev. Control 29 (2005) 57–70.
[30] W.S. Chang, I.H. Suh, T.W. Kim, Analysis and design of two types of digital repetitive control systems, Automatica 31 (1995) 741–746.
[31] S.S. Saab, Stochastic P-type/D-type iterative learning control algorithms, Int. J. Control 76 (2003) 139–148.
[32] T.H. Lee, K.K. Tan, S.Y. Lim, H.F. Dou, Iterative learning control of permanent magnet linear motor with relay automatic tuning, Mechatronics 10 (2000) 169–190.
[33] J.H. Lee, K.S. Lee, W.C. Kim, Model-based iterative learning control with a quadratic criterion for time-varying linear systems, Automatica 36 (2000) 641–657.
[34] K.S. Lee, J.H. Lee, I.S. Chin, H.J. Lee, A model predictive control technique for batch processes and its application to temperature tracking control of an experimental batch reactor, AIChE J. 45 (1999) 2175–2187.
[35] Y.Q. Wang, D.H. Zhou, F.R. Gao, Iterative learning model predictive control for multi-phase batch processes, J. Process Control 18 (2008) 543–557.
[36] K.S. Lee, J.H. Lee, Iterative learning control-based batch process control technique for integrated control of end product properties and transient profiles of process variables, J. Process Control 13 (2003) 607–621.
[37] K.S. Lee, J.H. Lee, Convergence of constrained model-based predictive control for batch processes, IEEE Trans. Automat. Control 45 (2000) 1928–1932.
[38] F. Xia, Y. Sun, NN-based iterative learning control under resource constraints: a feedback scheduling approach, ISNN 2005, LNCS 3498, 2005, pp. 1–6.

[39] H.J. Park, H.S. Cho, A fuzzy rule-based iterative learning control method with application to hydroforming processes, Mechatronics 2 (1992) 577–593.
[40] Y.Q. Wang, D.H. Zhou, F.R. Gao, Iterative learning reliable control of batch processes with sensor faults, Chem. Eng. Sci. 63 (2008) 1039–1051.
[41] S. Gunnarsson, M. Norrlöf, E. Rahic, M. Özbek, On the use of accelerometers in iterative learning control of a flexible robot arm, Int. J. Control 80 (2007) 363–373.
[42] A. Tayebi, J.-X. Xu, Observer-based iterative learning control for a class of time-varying nonlinear systems, IEEE Trans. Circuits Syst. I: Fundam. Theory Appl. 50 (2003) 452–455.
[43] S.S. Saab, A stochastic iterative learning control algorithm with application to an induction motor, Int. J. Control 20 (2004) 144–163.
[44] M. Arif, T. Ishihara, H. Inooka, Prediction-based iterative learning control (PILC) for uncertain dynamic nonlinear systems using system identification technique, J. Intell. Robot. Syst. 27 (2000) 291–304.
[45] H.K. Khalil, Nonlinear System, third ed., Prentice-Hall, New Jersey, 2002.
[46] Y.-Y. Tzou, S.-L. Jung, H.-C. Yeh, Adaptive repetitive control of PWM inverters for very low THD AC-voltage regulation with unknown loads, IEEE Trans. Power Electron. 14 (1999) 973–981.
[47] C.A. Desoer, M. Vidyasagar, Feedback Systems: Input–Output Properties, Academic Press, New York, 1975.
[48] S. Hara, Y. Yamamoto, T. Omata, M. Nakano, Repetitive control system: a new type servo system for periodic exogenous signals, IEEE Trans. Automat. Control 33 (1988) 659–668.
[49] J. Li, T.-C. Tsao, Robust performance repetitive control systems, J. Dynam. Syst. Meas. Control 123 (2001) 330–337.
[50] J. Wang, Q.P. He, S.J. Qin, C.A. Bode, M.A. Purdy, Recursive least squares estimation for run-to-run control with metrology delay and its application to STI etch process, IEEE Trans. Semicond. Manuf. 18 (2005) 309–319.
[51] C. Zhang, H. Deng, J.S. Baras, Run-to-run control methods based on the DHOBE algorithm, Automatica 39 (2003) 35–45.
[52] C.C. Palerm, L. Jovanovic, H. Zisser, F.J. Doyle III, W.C. Bevier, Prandial insulin dosing using run-to-run control, Diabetes Care 30 (2007) 1131–1136.
[53] G. Francois, B. Srinivasan, D. Bonvin, Convergence analysis of run-to-run control for a class of nonlinear systems, in: Proceedings of the American Control Conference, Denver, Colorado, 2003, pp. 3032–3037.
[54] K.J. Mclaughlin, Developing of techniques for real time monitoring and control in plasma etching, Ph.D. dissertation, University of Texas at Austin, 1989.
[55] A. Tayebi, C-J. Chien, A unified adaptive iterative learning control framework for uncertain nonlinear systems, IEEE Trans. Automat. Control 52 (2007) 1907–1913.
[56] C.-J. Chien, A. Tayebi, Further results on adaptive iterative learning control of robot manipulators, Automatica 44 (2008) 830–837.
[57] G.M. Bone, A novel iterative learning control formulation of generalized predictive control, Automatica 31 (1995) 1483–1487.
[58] P. Jiang, L.C.A. Bamforth, Z. Feng, J.E.F. Baruch, Y.Q. Chen, Indirect iterative learning control for a discrete visual servo without a camera-robot model, IEEE Trans. Syst. Man Cybern. Part B: Cybern. 37 (2007) 863–876.
[59] K.K. Tan, S. Zhao, J.-X. Xu, Online automatic tuning of a proportional integral derivative controller based on an iterative learning control approach, IET Control Theory Appl. 1 (2007) 90–96.
[60] Y.-Y. Tzou, R.-S. Ou, S.-L. Jung, M.-Y. Chang, High-performance programmable AC power source with low harmonic distortion using DSP-based repetitive control technique, IEEE Trans. Power Electron. 12 (1997) 715–725.
[61] C.-L. Chen, G.T.C. Chiu, Spatially periodic disturbance rejection with spatially sampled robust repetitive control, J. Dyn. Syst. Meas. Control 130 (2008) 1–11.
[62] R. Sreenivasan, T. Gougousi, Y. Xu, J. Kidder Jr., E. Zafiriou, G.W. Rubloff, Run to run control in tungsten chemical vapor deposition using H_2/WF_6 at low pressures, J. Vac. Sci. Technol. B 19 (2001) 1931–1941.
[63] R. Ganesan, T.K. Das, K.M. Ramachandran, A multiresolution analysis – assisted reinforcement learning approach to run-by-run control, IEEE Trans. Automat. Sci. Eng. 4 (2007) 182–193.
[64] M. Hankinson, T. Vincent, K.B. Irani, P.P. Khargonekar, Integrated real-time and run-to-run control of etch depth in reactive ion etching, IEEE Trans. Semicond. Manuf. 10 (1997) 121–130.
[65] J. Busch, A. Cruse, W. Marquardt, Run-to-run control of membrane filtration processes, AIChE J. 53 (2007) 2316–2328.
[66] T. Sugie, F. Sakai, Noise tolerant iterative learning control for a class of continuous-time systems, Automatica 43 (2007) 1766–1771.
[67] K.L. Moore, Iterative learning control – an expository overview, Appl. Comput. Control Signal Process. Circuits 1 (1999) 151–241.
[68] H.-S. Ahn, K.L. Moore, Y. Chen, Iterative Learning Control: Robustness and Monotonic Convergence for Interval Systems, Springer-Verlag, London, 2007.
[69] Y. Wang, H. Zisser, E. Dassau, L. Jovanovič, F.J. Doyle III, Model predictive control with learning-type set-point: application to artificial pancreatic β-cell, AIChE J., in press.
[70] Y. Wang, E. Dassau, F.J. Doyle III, Closed-loop control of artificial pancreatic β-cell in type 1 diabetes mellitus using model predictive iterative learning control, IEEE Trans. Biomed. Eng., in press, doi:10.1109/TBME.2009.2024409.
[71] C.C. Palerm, H. Zisser, L. Jovanovič, F.J. Doyle III, A run-to-run control strategy to adjust basal insulin infusion rates in type 1 diabetes, J. Process Control 18 (2008) 258–265.
[72] H. Dou, K.K. Tan, T.H. Lee, Z. Zhou, Iterative learning feedback control of human limbs via functional electrical stimulation, Control Eng. Pract. 7 (1999) 315–325.
[73] F. Doyle, L. Jovanovič, D. Seborg, R.S. Parker, B.W. Bequette, A.M. Jeffrey, X. Xia, I.K. Craig, T. McAvoy, A tutorial on biomedical process control, J. Process Control 17 (2007) 571–594.
[74] E. Del Castillo, J.-Y. Yeh, An adaptive run-to-run optimizing controller for linear and nonlinear semiconductor processes, IEEE Trans. Semicond. Manuf. 11 (1998) 285–295.
[75] J.-X. Xu, Z. Qu, Robust iterative learning control for a class of nonlinear systems, Automatica 34 (1998) 983–988.
[76] M. Heertjes, T. Tso, Nonlinear iterative learning control with applications to lithographic machinery, Control Eng. Pract. 15 (2007) 1545–1555.
[77] J.Y. Choi, H.J. Park, Use of neural networks in iterative learning control systems, Int. J. Syst. Sci. 31 (2001) 1227–1239.
[78] J.-X. Xu, B. Viswanathan, Adaptive robust iterative learning control with dead zone scheme, Automatica 36 (2000) 91–99.
[79] B.J. Driessen, N. Sadegh, Convergence theory for multi-input discrete-time iterative learning control with Coulomb friction, continuous outputs, and input bounds, Int. J. Adapt. Control Signal Process 18 (2004) 457–471.
[80] H.-F. Chen, H.-T. Fang, Output tracking for nonlinear stochastic systems by iterative learning control, IEEE Trans. Automat. Control 49 (2004) 583–588.