Recent Advances in Iterative Learning Control

Jian-Xin XU

(*Department of Electrical and Computer Engineering, National University of Singapore,*
4 *Engineering Drive* 3, 117576 *Singapore*)
(E-mail: elexujx@nus.edu.sg)

Abstract In this paper we review the recent advances in three sub-areas of iterative learning control (ILC): 1) linear ILC for linear processes, 2) linear ILC for nonlinear processes which are global Lipschitz continuous (GLC), and 3) nonlinear ILC for general nonlinear processes. For linear processes, we focus on several basic configurations of linear ILC. For nonlinear processes with linear ILC, we concentrate on the design and transient analysis which were overlooked and missing for a long period. For general classes of nonlinear processes, we demonstrate nonlinear ILC methods based on Lyapunov theory, which is evolving into a new control paradigm.

Key words Iterative learning control, linear processes, nonlinear processes

1 Linear ILC for linear systems

1.1 Why ILC

Consider a control task that requires the perfect tracking of a pre-specified target trajectory, e.g. track following of a hard disk drive, or temperature control of wafer process. The common features of this class of control problems are 1) task must be finished in a finite duration ranging from milliseconds to days, 2) the target trajectory must be strictly followed from the very beginning of the execution, and 3) the task is repeated from trial to trial, from batch to batch, or in general from iteration to iteration, under the same condition. We face a new class of control tasks: perfect tracking in a finite interval under a repeatable control environment.

Most existing control methods including adaptive or robust control, are not suitable for such class of tasks because of two reasons. First, these control methods are characterized by the asymptotic convergence, thus unable to achieve a perfect tracking even if the initial discrepancy is zero. Second and more important, they are not able to "learn" from previous task execution, whether succeeded or failed. Without learning, a control system can only produce the same performance without improvement even the task is repeated once again. Iterative Learning Control (ILC) was proposed to meet this kind of control requirements[1,2]. The idea of ILC is straightforward: use control information of the preceding execution to improve the present execution. This is realized through memory based learning.

ILC controllers can be constructed in many different ways. In this section we demonstrate four representative and most commonly used configurations. In general, ILC structures can be classified into two major categories: embedded and cascaded. In the following, the first three belong to embedded structure, and the fourth one belongs to the cascaded structure.

1.2 Previous cycle learning

The configuration of a previous cycle learning (PCL) scheme is shown in Fig. 1. Here the subscript i denotes the i-th iteration. Hence $y_{d,i}$, y_i, u_i and e_i denote the reference signal, output signal, control signal, and error signal respectively at the i-th iteration. G_p and G_{ff} denote the transfer functions of the plant and the control compensator, respectively. In cases the system perform the same control task, $y_{d,i+1} = y_{d,i}$. The MEM labeled with y, y_d and u are memory arrays storing system signals of the current cycle, *i.e.* $(i+1)$-th iteration, which will be used in the next learning cycle (iteration).

According to the PCL configuration shown in Fig. 1,

$$y_i = G_p u_i, \quad e_i = y_d - y_i, \quad u_{i+1} = u_i + G_{ff} e_i \tag{1}$$

Equation (1) is the PCL updating law. It is called previous cycle learning simply because only the previous cycle control signals u_i and error signals e_i are used to form the current cycle control input u_{i+1}. It is an open-loop control in the time domain, but a closed-loop control in the iteration domain.

Received January 8, 2004; in revised form July 5, 2004

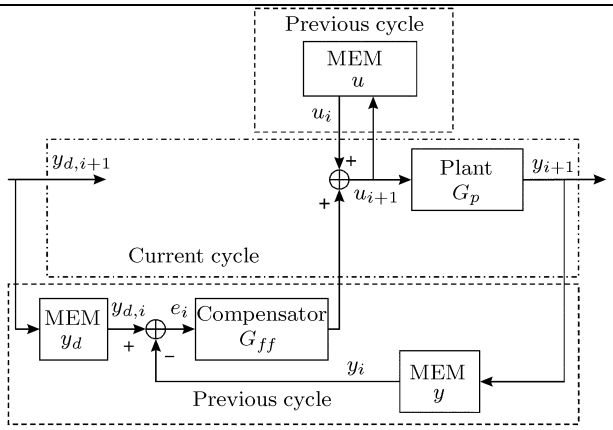

Fig. 1 The schematics of PCL

The learning convergence condition for PCL can be derived below,

$$e_{i+1} = y_d - y_{i+1} = y_d - G_p u_{i+1} = y_d - G_p(u_i + G_{ff}e_i) = (1 - G_p G_{ff})e_i \Rightarrow$$
$$\frac{e_{i+1}}{e_i} = 1 - G_p G_{ff} \Rightarrow \|\frac{e_{i+1}}{e_i}\| = \|1 - G_p G_{ff}\| \leqslant \gamma < 1 \tag{2}$$

where $\|G\| = |G(j\omega)|$ denotes the magnitude of the transfer function at a specified frequency ω. The norm $\|\cdot\|$ is defined as infinity norm for all frequencies $\leqslant \omega_b$ and ω_b is the bandwidth that may apply to any control scheme. Clearly, as far as the tracking error signals of the 1st iteration, e_0, is finite, we have $\|e_i\| \leqslant \gamma^i \|e_0\| \to 0$ as $i \to \infty$. It can also be seen that, for a specified threshold of the tracking error, a smaller initial error profile e_0 may expedite the learning process.

1.3 Current cycle learning

Due to the open-loop nature, PCL could be sensitive to small perturbations. This can be improved by a feedback based learning if the loop can be closed appropriately, leading to the current cycle control (CCL). The configuration of the CCL scheme is shown in Fig. 2.

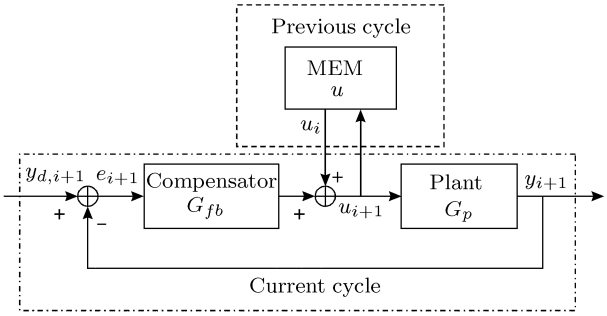

Fig. 2 The schematics of CCL

Accordingly, the updating law of the CCL scheme is

$$u_{i+1} = u_i + G_{fb}e_{i+1},$$

where the G_{fb} is the transfer function of the compensator which is in fact a feedback controller. It is called current cycle learning because the current cycle tracking error, e_{i+1}, is involved in learning.

The convergence condition for CCL is derived as follows

$$e_{i+1} = y_d - y_{i+1} = y_d - G_p u_{i+1} = y_d - G_p(u_i + G_{fb}e_{i+1}) = y_d - G_p u_i - G_p G_{fb} e_{i+1}$$
$$(1 + G_p G_{fb})e_{i+1} = e_i \Rightarrow \frac{e_{i+1}}{e_i} = \frac{1}{1 + G_p G_{fb}} \Rightarrow \|\frac{e_{i+1}}{e_i}\| = \|\frac{1}{1 + G_p G_{fb}}\| \leqslant \gamma < 1 \tag{3}$$

It can be seen that PCL and CCL are functioning in a complementary manner. PCL requires the factor $\|1 - G_pG_{ff}\|$ below 1 for all frequencies within the band $[0, \omega_b]$, and often leads to a low gain G_{ff}. CCL requires the factor $\|1 + G_pG_{fb}\|$ above 1 for all frequencies within the band $[0, \omega_b]$, and often leads a high gain G_{fb}. Note that the memory arrays required for CCL are half of the PCL.

1.4 Previous and current cycle learning

Generally speaking, it may be a difficult job to find a suitable G_{ff} such that the (2) is satisfied, *i.e.*, $\|1 - G_pG_{ff}\|$ is strictly less than 1 for all frequencies in $[0, \omega_b]$. Likewise, it may be a difficult job to find a suitable G_{fb} such that the denominator in (3) is strictly larger than 1 for all frequencies in $[0, \omega_b]$. There is only 1 degree of freedom (DOF) in the controller design for both schemes. By pairing PCL and CCL together, there is a possibility that the learning convergence will be improved. This can be achieved if the convergence conditions (2) and (3) can complement to each other at frequencies where one convergence condition is violated. There are several ways to pair the PCL and CCL, and a possible combination is shown in Fig. 3.

It can be easily derived that the convergence condition is

$$\frac{\|1 - G_pG_{ff}\|}{\|1 + G_pG_{fb}\|}$$

that is, the multiplication of both conditions of PCL and CCL. To demonstrate the advantage of the 2 DOF design with PCCL, an example is shown in Fig. 4, where $1 - G_pG_{ff} = \dfrac{s^2 - 3}{3s^2 + 3s + 5}$ and $1/(1 + G_pG_{fb}) = \dfrac{s^2 + 20s + 80}{2s^2 + 10s + 130}$. Note that neither the PCL condition (2) nor the CCL condition (3) holds for all frequencies, but the convergence condition with the integrated PCCL, which is the superposition of PCL and CCL in Bode plot, does hold for all frequencies.

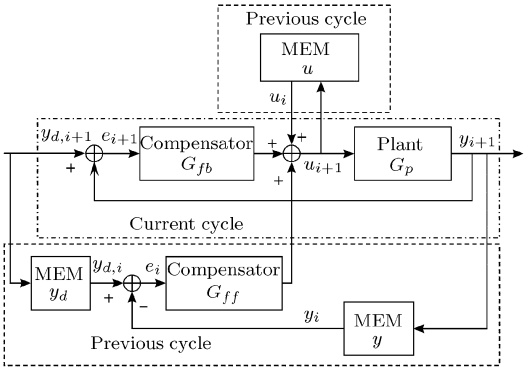

Fig. 3 The schematics of PCCL

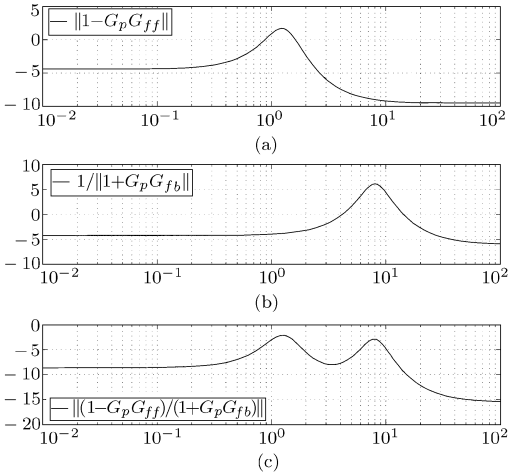

Fig. 4 The complementary role of PCL and CCL

1.5 Cascaded ILC

In the previous three ILC schemes, one may observe that the control system has been redesigned, with a new feedforward component added to the system input channel. From the configurations shown in Fig. 1, 2 and 3, a new control block is embedded into the control loop. Such an embedded structure is the common structure for most existing ILC schemes. Hence, if an ILC mechanism is to be incorporated into an existing control system, either the core execution programme needs to be rewritten or the micro-controller chip needs to be replaced. In many real applications, such a re-configuration of a commercial controller is not acceptable due to the cost, security and intellectual property problems. For instance, a rapid thermo-processing device in wafer industry costs millions of dollars, and the only tunable part is a number of set-points. In such circumstance, the cascaded learning method is suitable as it modifies only the reference trajectory iteratively to improve the control performance.

The schematics of such an ILC is demonstrated in Fig. 5.

It can be seen that the ILC block is "cascaded" to the existing control loop. The ILC with cascaded structure will use the modified reference signals and the actual system output of previous cycle to generate the new reference signals for the current cycle. Owing to the cascaded structure, the ILC need not be embedded into the existing control loop, thus avoids any reconfiguration of the system hardware. What is needed is essentially some re-programming of reference signals, which can be easily carried out in real applications.

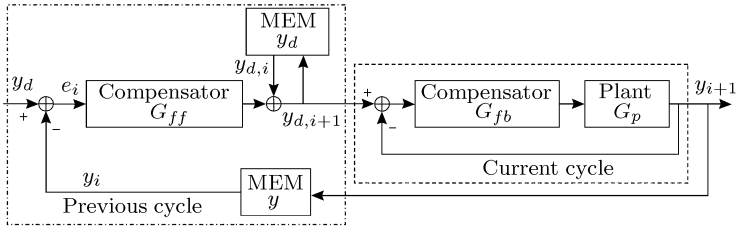

Fig. 5 The schematics of a cascaded structure ILC

According to Fig. 5, we have

$$y_i = Gy_{d,i}, \quad e_i = y_d - y_i, \quad y_{d,i+1} = y_{d,i} + G_{ff}e_i, \quad y_{d,0} = y_d \tag{4}$$

where $G = \dfrac{G_pG_{fb}}{1 + G_pG_{fb}}$ denotes the closed-loop transfer function; y_d is the original reference repeated over a fixed operation period; $y_{d,i}$ is the reference signal modified via learning for the control loop. According to the learning control law (4), the convergence condition for the cascaded structure ILC can be derived as

$$\begin{aligned} e_{i+1} &= y_d - y_{i+1} = y_d - Gy_{d,i+1} = y_d - G(y_{d,i} + G_{ff}e_i) = y_d - Gy_{d,i} - GG_{ff}e_i = (1 - GG_{ff})e_i \\ &\Rightarrow \frac{e_{i+1}}{e_i} = 1 - GG_{ff} \Rightarrow \|\frac{e_{i+1}}{e_i}\| = \|1 - \frac{G_pG_{fb}G_{ff}}{1 + G_pG_{fb}}\| \leqslant \gamma < 1 \end{aligned}$$

In most cases the cascaded ILC is of PCL type, because set points, once selected, cannot be changed in the midst of real-time operation.

2 Linear ILC for nonlinear systems

In this section, we consider linear ILC schemes for nonlinear dynamical systems. Consider the following dynamical systems

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), u(t), t), \quad \boldsymbol{x}(0) = \boldsymbol{x}_0, \quad y(t) = g(\boldsymbol{x}(t), u(t), t) \tag{5}$$

where $t \in [0,\ T]$, $\boldsymbol{x}(t) \in R^n$, $y(t) \in \mathcal{R}$ and $u \in \mathcal{R}$, $\boldsymbol{f}(\cdot)$ is a smooth vector field, and $g(\cdot)$ is a smooth function.

Under the repeatability of the control environment, the control objective for ILC is to design a sequence of appropriate control inputs $u_i(t)$ such that the system output $y_i(t)$ approaches the target trajectory $y_d(t)$, $\forall t \in [0,\ T]$. In Arimoto's first ILC article[1] a typical yet the simplest linear-type first-order iterative learning control scheme is proposed

$$u_{i+1}(t) = u_i(t) + \beta\Delta y_i(t) \tag{6}$$

where $\Delta y_i = y_d - y_i$, and β is a constant learning gain. The initial control profile $u_0(t)$, $t \in [0,\ T]$ is either set to zero or initialized appropriately by some control mechanism. The convergence condition is determined by the relation

$$|u_{i+1}|_\lambda \leqslant |1 - \beta g_u||u_i|_\lambda, \quad |1 - \beta g_u| = \gamma < 1 \tag{7}$$

where $g_u(\boldsymbol{x},u,t)=\dfrac{\partial g}{\partial u}$, and $|\cdot|_\lambda$ is the time weighted norm defined as $\max\limits_{t\in[0,T]} e^{-\lambda t}|\cdot|$. The controllability condition is that the system gain $g_u\in[\alpha_1,\ \alpha_2]$, either $\alpha_1>0$ or $\alpha_2<0$. By choosing a proper learning gain β, the convergence condition (7) can be fulfilled if the interval $[\alpha_1,\ \alpha_2]$ is known *a priori*.

There are two conditions indispensable in linear-type ILC: the global Lipschitz continuity condition (GLC) and resetting condition, which are summarized below.

1) $\boldsymbol{f}(\boldsymbol{x},u,t)$ and $g(\boldsymbol{x},u,t)$ are global Lipschitz continuous with respect to $\boldsymbol{x}$ and u.

2) identical initialization condition (*i.i.c.*) $y_i(0)=y_d(0)$.

Mathematically we know that GLC guarantees the existence and uniqueness of the differential Equation (5) for any initial value. In ILC theory, the importance of GLC lies in that there is no finite escape time if input is finite. Therefore a typical iterative learning process, characterized as open-loop control, can be carried on in a finite interval without concerning time domain stability or convergence.

The identical initialization condition, one fundamental assumption of ILC, has been criticized by control experts from different disciplines. This condition, however, cannot be relaxed if the perfect tracking is pursued.

2.1 Robust optimal design

The popularity of H_∞ control is in large owing to its systematic robust optimal design. A pure robust design may give too much weight on stability issue and sacrifice the performance. On the contrary, a pure optimal design could be sensitive to system uncertainties. We need a robust optimal design for ILC, wherever applicable. The most challenging task is, can we design a robust optimal ILC for highly nonlinear systems such as (5)? Any control theory would not be complete without a systematic design. On the other hand, it is perhaps one of the hardest jobs for nonlinear uncertain systems. The task is highly related to the selection of performance index or the cost function, and types of system uncertainties. Here we will show one such robust optimal ILC design for the system (5), which is to maximize the convergence speed under an interval uncertainty[3]. The ILC design can be formulated into a min-max optimization problem with a systematic solution. The *max* operation is to maximize the influence from the system uncertainty, and *min* operation is to minimize a learning factor that determines the convergence speed.

The convergence speed is determined by the relation

$$|u_{i+1}|_\lambda\leqslant\gamma|u_i|_\lambda \tag{8}$$

and the slowest one is given by $|u_{i+1}|_\lambda=\gamma|u_i|_\lambda$. Let us define a "characteristic equation" in the iteration domain that specifies the convergence speed

$$z-\gamma=z-|1-\beta g_u|=0 \tag{9}$$

The smaller the γ, the faster the convergence speed. However the value of γ depends on the unknown system gain g_u which could be nonlinear, uncertain and varying in the interval $\mathcal{D}=[\alpha_1,\ \alpha_2]$. The learning gain β should be designed in such way that γ is the lowest in the presence of the interval uncertainty.

This problem can be formulated mathematically as the following min-max problem:

$$J=\min_{\beta\in R}\max_{g_u\in\mathcal{D}}\ |1-\beta g_u|,\quad \mathcal{D}=[\alpha_1,\ \alpha_2]. \tag{10}$$

Note that the min-max operation achieves the robust optimal design, as the *min* operation realizes the fastest convergence speed by minimizing γ, and the *max* operation considers the worst case learning convergence by maximizing the influence from g_u.

To facilitate discussion for various ILC schemes, consider the following robust optimization problem which includes (10) as a special case:

$$J=\min_{\beta\in R}\max_{g_u\in\mathcal{D}}\ |p-\beta g_u|,\quad p\in\mathcal{R} \tag{11}$$

The above learning convergence factor $|p-\beta g_u|$ is corresponding to the following iterative learning scheme

$$u_{i+1}(t)=pu_i(t)+\beta\Delta y_i(t) \tag{12}$$

with p a weighting factor in general, and a forgetting factor in particular when $p \in (0, 1)$. The solution of (11) is given in the following Proposition.

Proposition 1[3]. When $\beta = \frac{2p}{\alpha_2 + \alpha_1}$, $J = \min\limits_{\beta \in R} \max\limits_{g_u \in \mathcal{D}} |p - \beta g_u|$ reaches its minimum

$$\gamma^* = |p| \frac{\alpha_2 - \alpha_1}{\alpha_2 + \alpha_1} \tag{13}$$

γ^* gives the fastest convergence response of ILC. To achieve perfect learning, it is necessary that $p = 1$. Consequently $\gamma^* = \frac{\alpha_2 - \alpha_1}{\alpha_2 + \alpha_1}$. Most existing ILC schemes deal with the interval uncertainty $g_u \in [\alpha_1,\ \alpha_2]$ in a conservative manner by setting (assume $\alpha_1 > 0$)

$$\beta = \frac{1}{\alpha_2}$$

In such case the range of the convergence factor is

$$0 \leqslant |1 - \beta g_u| \leqslant \frac{\alpha_2 - \alpha_1}{\alpha_2}$$

It is immediately obvious that

$$\gamma^* = \frac{\alpha_2 - \alpha_1}{\alpha_2 + \alpha_1} < \frac{\alpha_2 - \alpha_1}{\alpha_2}$$

Although both designs consider the worst case, the robust optimal design can reduce the convergence factor by 50% in the extreme case $\alpha_1 \to \alpha_2$.

2.2 Transient analysis

ILC achieves the perfect tracking over the interval $[0,\ T]$, namely achieves a perfect transient response in the time domain. However, it may encounter another transient behavior along the iteration axis. It is well known that a convergent sequence may produce an unacceptable transient response. A numerical example[4] demonstrates that a convergent sequence in the iteration domain could have the worst case error bound above 100^{200}!

In classical control, the transient performance are evaluated in terms of settling time, overshoot, critical/overdamped or oscillatory response. In iteration domain it is necessary to quantify the transient performance in a similar way. It was proposed to evaluate an iterative process in terms of the convergence speed, the global uniform bound (maximum tracking error), and the maximum monotonic convergence interval[3]. Generally speaking, it is very difficult to evaluate the transient behavior of a nonlinear process, and it is even tougher for ILC because of the performance along the time axis and iteration axis.

The convergence speed has been partially solved by introducing a new metric – Q-factor which is originally used to evaluate iteration algorithms in numerical analysis[5]. By means of the Q-factor, together with the robust optimal design, we are able to quantify the learning convergence speed of various ILC schemes. Let us briefly mention two such results associated with Q-factor.

First is concerned with a controversial issue: can a high order ILC scheme outperform a lower order ILC. Ever since the first article regarding high order ILC[6], it was acknowledged that a high order ILC can improve learning speed. Intuitively, a higher order ILC, that employs preceding control information of more than one iteration, should be able to improve learning performance as more of preceding control information is used. However, a simple *linear* combination of preceding control information may not be able to generate or provide anything new information. What is more, for a convergent ILC sequence, in most iterations the latest should be the most accurate and the rest are less. A linear combination of less accurate ones seems only to degrade the performance.

Later it was found that the solution of a 2nd order ILC discussed in[6] does not exist. It was still unclear about the general situations until the concept of Q-factor and the min-max design were proposed. The conclusion[3] was, the convergence speed of lower order ILC is always faster than that of higher order ILC in terms of the time weighted norm.

The second is concerned with three ILC algorithms of the linear, Secant and Newton types. Quantified evaluation based on Q-factor tells us that, by introducing a nonlinear gain such as the Secant or Newton type ILC, the learning convergence can be greatly expedited[4].

Another hot topic of ILC is: can an ILC algorithm warrant a monotonic convergence along the iteration axis. Without such a warranty, one cannot help worrying about whether the tracking error

could reach a level of 10^{200} during the transient period. Transient behavior is an open problem not only for ILC, but also for most control methods, and is difficult to analyze even for linear systems.

Recently the time domain dynamical impact to the iteration process was exploited[3]. As far as a PCL type ILC is concerned, the time domain stability is not guaranteed because of the open-loop nature. In such circumstance, a divergent dynamics can be suppressed by the time weighted norm which has an attenuation factor along the time axis, $e^{-\lambda t}$, if λ is sufficiently large.

A few questions naturally arise. First, λ is neither a system parameter nor a design parameter, is the learning convergence really depending on it? Second, what may happen if a small λ is considered for the time weighted norm, or in other words, when the dynamic impact is not ignored? Third, how large could be the dynamic impact to the learning convergence, can we really ignore it? Most ILC works end by assuming a sufficiently large λ, which results in a monotonic convergence in the time weighted norm. In order to make clear the underlying relationship between λ and system dynamical behavior, some work was done with the objective to quantify λ, which leads to several interesting findings[4].

If we use $|\cdot|_\lambda$ with a sufficiently large λ as a yardstick to measure the learning process, ILC is guaranteed to converge *monotonically* over the entire time interval $[0,\ T]$. If however a smaller λ is employed, ILC can only guarantee the monotonic convergence over a subinterval $[0,\ T_1] \subset [0,\ T]$ according to such a norm $|\cdot|_\lambda$. A smaller λ indicates a more restrict assessment to the system performance. It is possible to compute the minimum λ required to generate a monotonic convergence over $[0,\ T]$ assessed by $|\cdot|_\lambda$. On the other hand, consider the extreme case where $\lambda \to 0$, which in fact leads to the supreme norm. It is able to estimate the maximum interval $[0,\ T_1]$ on which the learning converges monotonically in the supreme norm $|\cdot|$.

Looking at the relationship between the two norms $|\cdot|_\lambda$ and $|\cdot|$, and the minimum λ, the maximum tracking error bound can be estimated by

$$|\cdot| \leqslant |\cdot|_\lambda e^{\lambda T}$$

2.3 Old problems and new solutions

The GLC condition and the identical initialization condition are two fundamental postulates which lay the foundation for the entire ILC framework. These two conditions, however, greatly confine the applicability of ILC, often incur criticism. Can we remove these two conditions? The answer is no for classical ILC based on contractive mapping. GLC prevents the control system from the finite escape time phenomenon, and *i.i.c.* ensures that the learning convergence can be achieved from the very beginning. Counterexamples were given to show the divergent response without these two conditions[4].

Classical ILC methods, no matter PCL, CCL or PCCL, all belong to contractive mapping. The ILC law links two iterations and results in a contractive mapping between two consecutive iterations as

$$\frac{|e_{i+1}|}{|e_i|} \leqslant \gamma < 1$$

Classical ILC does not use the system dynamical knowledge such as $\boldsymbol{f}(\boldsymbol{x}, u, t)$, and does not bother to design a stable closed-loop in the time domain. In fact, if we restrict ourselves to the present ILC framework, the asymptotic convergence along the iteration axis in $|\cdot|_\lambda$ is perhaps the best we can expect. Look at the linear-type ILC schemes (6), which is almost model free. The only system knowledge used is $g_u \in \mathcal{D}$, regardless of the highly nonlinear, non-affine and uncertain dynamics. It also focuses on one of the most difficult control tasks: perfect tracking over the entire time interval $[0,\ T]$ using only the static output control. We cannot be too demanding with so less system information and so general a control objective.

The only way to relax the two postulates is to incorporate more of system knowledge in ILC design. Comparatively GLC is easier to be generalized, and *i.i.c.* is related to the system repeatability which is more fundamental. In next Section we demonstrate how to design a nonlinear ILC by fully making use of the system dynamical knowledge, in the sequel successfully deal with local Lipschitzian functions.

3 Nonlinear ILC for nonlinear systems

Consider a dynamics with local Lipschitzian nonlinearity

$$\dot{x} = \theta(t)x^2 + u, \quad x(0) = x_0 \tag{14}$$

where $\theta(t) \in \mathcal{C}[0,\ T]$.

Let the target trajectory be $x_d(t)$, the nonlinear iterative control law is

$$u_i = ke_i + \dot{x}_d - \hat{\theta}_i(t)x_i^2 \tag{15}$$

and the nonlinear parametric learning law is $\forall t \in [0,\ T]$

$$\hat{\theta}_i(t) = \hat{\theta}_{i-1}(t) - x_i^2(t)e_i(t), \quad \hat{\theta}_{-1}(t) = 0 \tag{16}$$

where $e_i = x_d - x_i$.

To analyze the learning convergence, the following Lyapunov functional is used

$$V_i(t) = \frac{1}{2}e_i^2(t) + \frac{1}{2}\int_0^t \phi_i^2(\tau)\mathrm{d}\tau. \tag{17}$$

where $\phi_i = \theta - \hat{\theta}_i$. It can be proven[7] that $e_i(t)$ converges to zero pointwisely as $i \to \infty$.

In the following we will briefly demonstrate several new features achieved by this class of nonlinear ILC.

3.1 Quasi-optimal ILC

Aiming at balancing control performance vs control effort, nonlinear optimal control has been the active subject of considerable research work over the past few decades[8]. By applying the standard dynamic programming, the optimal control can be converted to the problem of solving partial differential equation known as Hamilton-Jacobi-Bellman (HJB) equation. There are however two obstacles in achieving optimal ILC. First, it is difficult to find the closed form optimal control for general nonlinear systems because of the difficulty in solving the nonlinear partial HJB differential equation. Second, ILC *by default* is supposed to handle systems with uncertainties.

To avoid the first obstacle, a suboptimal control strategy based on control Lyapunov function and Sontag's formula[9] is used, which provides a suboptimal performance as well as stability along time horizon for a broad class of nonlinear dynamic systems. Regarding the second obstacle, we assume that the system dynamics can be separated into a nominal part and an uncertain part as below

$$\dot{x}_i = f_i + \theta(t)x_i^2 + u_i \tag{18}$$

where $f_i = f(x_i, t)$ is a known smooth function, and considered as the nominal part. The suboptimal control law will be designed based on the system nominal part

$$\dot{x}_i = f_i + u_i \tag{19}$$

and the ILC with pointwise adaptation mechanism is to address the uncertain part $\theta(t)x_i^2$ as before.

It should be noted that f_i can be easily canceled out by incorporating a term $-f_i$ in the system input u_i. Cancelation is nevertheless a passive way whereas optimal design is an active way of making use of the system knowledge f_i. For example, if $f_i = Ax_i$, we can construct a linear quadratic optimal controller accordingly. Besides, in many practical systems the nominal part consists of either linear or nonlinear damping term, which is a stabilizing force. If f_i is much smaller than $\theta(t)x_i^2$, the effect of optimality may be minor. Indeed, if the system is predominant by the uncertain part, other control methods such as robust control, adaptive control or ILC instead of optimal control should be used. On the contrary, if the nominal part dominates, the effect of optimality becomes obvious. Often we cannot tell which is dominant in practice, then it should be no harm to let ILC and optimal control coexist.

Consider a particular case $f_i = -x_i^3$ which is a nonlinear damping. The error dynamics under iteration is

$$\dot{e}_i = \dot{x}_d + x_i^3 - \theta(t)x_i^2 - u_i \tag{20}$$

The nominal part of the error dynamics is $\bar{f}_i = \dot{x}_d + x_i^3$. The objective function of optimal control is

$$J = \inf_{u_i} \int_0^T [q(e_i) + u_i^2]\mathrm{d}t \tag{21}$$

If it is possible to solve the following HJB equation

$$q(e_i) - \frac{1}{4}(\frac{\partial V^*}{\partial e_i})^2 + \frac{\partial V^*}{\partial e_i}\bar{f}_i = 0 \tag{22}$$

to find the closed form of the value function V^*, the optimal control is given as

$$u^* = -\frac{1}{2}\frac{\partial V^*}{\partial e_i}$$

However it is not an easy job to solve the HJB equation. Hence the Sontag's formula is introduced to provide a suboptimal solution to the nonlinear system (20) as follows

$$u_{op,i} = \bar{f}_i + \sqrt{(\bar{f}_i)^2 + q(e_i)}sign(\frac{\partial V}{\partial e_i}) \tag{23}$$

where $sign(\cdot)$ is the signum function, and V is an arbitrary Lyapunov function (in general Control Lyapunov Function). It can be seen that the idea of the suboptimal control with Sontag's formula is to use a Lyapunov function V to replace the value function $V^* = \inf_{u_i}\int_t^{\mathrm{T}}[q(e_i) + u_i^2]\mathrm{d}t$ which is hard to solve from the HJB equation. It is also worth to note how the nonlinear nominal part $\bar{f}_i$ is incorporated in the suboptimal control law (23), which is not merely a simple cancelation.

Quasi-optimal ILC is achieved by combining the suboptimal control and ILC simply in an additive form,

$$u_i = u_{op,i} + u_{l,i}, \quad u_{l,i} = ke_i - \hat{\theta}_i(t)x_i^2, \quad \hat{\theta}_i(t) = \hat{\theta}_{i-1}(t) - x_i^2 e_i \tag{24}$$

where $u_{l,i}$ is the learning part with pointwise parametric adaptation.

The closed-loop performance will depend on the selection of the weighting function $q(e_i)$. A large $q(e_i)$ implies more penalty on the tracking error, consequently leads to a faster convergence at the price of large control signals. It offers extra design degree of freedom, in a sense like the LQR, although the control problem is far more difficult[10].

3.2 New findings

3.2.1 Learning from different reference trajectories

That the target trajectory must be uniformly identical for all iterations is one of the fundamental conditions, hence one of the fundamental constraints, for all kinds of ILC schemes. Now we move one step forward – let ILC mechanism learn from different target trajectories, that is, trajectories may vary from iteration to iteration. There were some pioneer work[11,12], which are in essence a Least Square approach. The limitation of those methods are the demand for accurate information of control input and output signals of previous trials. Can we let ILC start from *scratch* and converge even if the target trajectory $x_{d,i} \in C^1[0,\ T]$ may vary at each iteration?

If the *i.i.c.* condition is satisfied, *i.e.* $x_i(0) = x_{d,i}(0)$, this problem is rather straightforward under the framework of Lyapunov Functional[13]. The error dynamics is

$$\dot{e}_i = -\theta(t)x_i^2 + \dot{x}_{d,i} - u_i, \qquad e_i(0) = 0 \tag{25}$$

The only change is from the identical trajectory x_d to the iteration dependent $x_{d,i}$, which are nevertheless known to us. Accordingly the control law is

$$u_i = ke_i + \dot{x}_{d,i} - \hat{\theta}_i(t)x_i^2 \tag{26}$$

and the parametric updating law remains the same as (16).

Since $x_{d,i}$ is now iteration i-dependent, generally speaking it cannot be treated through learning. The pointwise adaptation, as a functional approximation process, works only for iteration i-independent functions, such as $\theta(t)$. This clearly shows the learnability of nonlinear ILC.

3.2.2 Relaxation of *i.i.c.*

Now we return to the most difficult issue: can we remove the identical initialization condition, which has been with us all the way. From the differential equation theory, the initial condition will determine the solution trajectory of a nonlinear dynamics. A tiny discrepancy in initial conditions may lead to completely different solutions. However, a perfect initial resetting requires that the control

system be equipped with a precise homing mechanism, which may not be possible for many practical engineering systems.

Note that in classical ILC, the control objective is output tracking and the state variables are assumed neither available nor manoeuvrable. In nonlinear ILC, however, we make full use of the system knowledge especially concerning state dynamics. This opens a new avenue: replacing the *i.i.c.* with a less restricted initial condition – alignment condition – and meanwhile achieving the convergent property[14]. The alignment condition is simply $x_i(0) = x_{i-1}(T)$, *i.e.* the end state of preceding iteration becomes the initial state of the present iteration. In addition to this, we also need $x_d(0) = x_d(T)$. The rationale of the new condition can be easily perceived: we restart from wherever we stopped at. In the sequel we can avoid doing the extra task of bringing the system back to a specific place.

Let us further exploit this key issue. The identical initialization condition in ILC usually implies both spatial resetting and temporal resetting. While time resetting is natural for a task to be finished and repeated over a finite period, the spatial resetting is however not an easy job and not so imperative. Note that it is the spatial resetting which gives rise to extra implementation difficulty and incurs criticism.

Consider a target trajectory $x_d(t) \in C^1[0,\ T]$, which forms a continously spatial path. When do we need the spatial resetting? It is necessary only when the spatial path of the target trajectory is not completely closed, *i.e.* $x_d(0) \neq x_d(T)$. For instance, $x_d(t) = t$, $t \in [0,\ 1]$. In such circumstance, a perfect tracking will lead to $x_i(T) = x_d(T) \neq x_d(0)$. Hence an independent control mechanism must work appropriately between two consecutive iterations so as to bring back the system state to the initial position $x_d(0)$.

For any trajectories spatially closed, *i.e.* $x_d(0) = x_d(T)$, we can use the alignment condition and remove the spatial resetting requirement, as discussed above.

3.2.3 Extention to repetitive tasks

By relaxing the spatial resetting to the alignment condition, we can now extend ILC to repetitive control tasks – either tracking a periodic trajectory or reject a periodic disturbance over $[0,\ \infty)$. Let us exhibit how to convert a repetitive control task into an ILC task[14]. Consider the target trajectory $x_d(t) \in C^1[0,\ \infty)$ with the periodicity $x_d(t) = x_d(t - T)$. Assume that $\theta(t)$ is also periodic with the same period T. Define the state $x_i(t) = x((i-1)T + t)$, $\forall i = 1, 2, \cdots$. By virtue of the continuity, $x(iT)$ is the end point of the i-th iteration defined over $[(i-1)T,\ iT]$, and also the initial point of the $(i+1)$-th iteration defined over $[iT,\ (i+1)T]$. Note that the alignment condition is met because $x_i(T) = x_{i+1}(0)$ is in fact the same point $x(iT)$. Thus the original control problem is equivalent for $x_i(t)$ to track $x_d(t)$ over the period $[0,\ T]$, and the ILC can be directly applied.

What can we gain by converting a repetitive control problem into ILC problem? First of all, ILC is now able to handle periodic signals defined in infinite horizon, hence cover repetitive control problems. Second, ILC based on Lyapunov functional is able to handle more general classes of system nonlinearities and uncertainties. Indeed, the convergence analysis of classical repetitive control is mainly based on small gain theorem, quite similar to the contraction mapping, consequently the application is rather limited.

4 Concluding remarks

ILC methods offer two degrees of freedom: one in time domain and another in iteration domain. A typical ILC evolving only in iteration domain avoids sophisticated feedback design and stability analysis in time domain, but the resulting ILC may not possess any robustness in time domain. By incorporating feedback in ILC, the learning process is more robust in time domain, however there is no warranty that iteration domain performance can also be improved by this feedback. The inherent relationship between two domains remains open for us to further explore.

Also, like other control methodologies, there are numerous problems open in ILC field. Some are concerned with the system geometric properties, such as the relative degrees, dissipativity, etc. Some are concerned with the system control, such as the optimality in both time and iteration domains. Some are concerned with implementation issues, such as measurement noise, sampling, input saturation, deadzone, etc. Some are concerned with the control tasks, such as non-uniform tracking, pseudo-periodicity, etc.

Despite all the difficulties, many researchers are fighting hard and developing new ILC approaches, such as backstepping[15,16], stochastic ILC[17,18], etc. ILC is a relatively new research area and can be easily integrated with any other control methods under the system repeatability. Hence the readers of this introductory article, once plowing in the ILC field, can expect harvest from two aspects: either incorporating the concept and methods of ILC into their own fields to improve the performance, or incorporating their favorite methods into ILC to come up with a new control method.

References

1 Arimoto A, Kawamura S, Miyazaki F. Bettering operation of robots by learning. *Journal of Robotic Systems*, 1984, **1**: 123∼140

2 Bien Z, Xu J X. Iterative Learning Control — Analysis, Design, Integration and Applications. Kluwer Academic Publishers. Boston, USA, 1998

3 Xu J X, Tan Y. Robust optimal design and convergence properties analysis of iterative learning control approaches. *Automatica*, 2002, **38**(11): 1687∼1880

4 Xu J X, Tan Y. Linear and Nonlinear Iterative Learning Control. *Lecture Notes in Control and Information Science*, **291**, Springer-Verlag, Germany, 2003

5 Ortega J M, Rheinboldt. Iterative Solutions of Nonlinear Equations in Several Variables. Academic Press, New York, USA, 1970

6 Bien Z, Huh K M. High–order iterative learning control algorithm. *IEE Proceedings, Part–D, Control Theory and Applications*, 1989, **136**: 105∼112

7 Xu J X, Tan Y. A composite energy function based learning control approach for nonlinear systems with time-varying parametric uncertainties. *IEEE Transactions on Automatic Control*, 2002, **47**: 1940∼1945

8 Sepulchre M J R, Kokotovic P V. Constructive Nonlinear Control. Springer-Verlag, London, 1997

9 Primbs J, Nevistic V, Doyle J. Nonlinear optimal control: A control Lyapunov function and receding horizon perspective. *Asian Journal of Control*, 1999, **1**: 14∼24

10 Xu J X, Tan Y. A suboptimal learning control scheme for nonlinear systems with time-varying parametric uncertainties. *Journal of Optimal Control – Applications and Theory*, 2001, **22**: 111∼126

11 Kawamura S, Miyazaki F, Arimoto S. Realization of robot motion based on a learning method. *IEEE Transactions on Systems, Man, and Cybernetics*, 1988, **18**: 126∼134

12 Xu J X, Zhu T. Dual-scale direct learning of trajectory tracking for a class of nonlinear uncertain systems. *IEEE Transactions on Automatic Control*, 1999, **44**: 1884∼1888

13 Xu J X, Xu J. On iterative learning for different tracking tasks in the presence of time-varying uncertaintie. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 2004, **34**(1): 589∼597

14 Xu J X, Badrinath V, Qu Z H. Robust learning control for robotic manipulators with an extension to a class of non-linear systems. *International Journal of Control*, 2000, **73**(10): 858∼870 (special issue on ILC)

15 Qu Z H, Xu J X. Asymptotic learning control for a class of cascaded nonlinear uncertain systems. *IEEE Transactions on Automatic Control*, 2002, **46**(8): 1369∼1376

16 Tian Y P, Yu X H. Robust learning control for a class of nonlinear systems with periodic and aperiodic uncertainties. *Automatica*, 2003, **39**: 1957∼1966

17 Saab S S. A discrete-time stochastic learning control algorithm. *IEEE Transactions on Automatic Control*, 2001, **46**(6): 877∼887

18 Chen H F, Fang H T. Output tracking for nonlinear stochastic systems by iterative learning. *IEEE Transactions on Automatic Control*, 2004, **49**(4): 583∼588

Jian-Xin XU Received his bachelor degree from Zhejiang University, P.R.China in 1982. He attended the University of Tokyo, Japan, where he received his master's and PhD degrees in 1986 and 1989 respectively. All his degrees are in electrical engineering. He worked for one year in the Hitachi research Laboratory, Japan; for more than one year in Ohio State University, U.S.A. as a visiting scholar; and for 6 months in Yale University as a visiting research fellow. In 1991 he joined the National University of Singapore, and is currently an associate professor in the Department of Electrical Engineering. His research interests lie in the fields of learning control, variable structure control, fuzzy logic control, discontinuous signal processing, and applications to motion control and process control problems. He has accomplished 12 research projects. He is a senior member of IEEE.

Up to now he produced 100 peer-reviewed journal papers, in which half are in IEEE transactions, Automatica and SIAM. He has one monograph: "Linear and Nonlinear Iterative Learning Control", by Springer-Verlag in 2003, and co-edited 3 books: ≪Iterative Learning Control≫ published by Kluwer Academic Press in 1998, ≪Advances in Variable Structure Systems≫ by World Scientific in 2000, and ≪Variable Structure Systems: Towards the 21st Century≫, by Springer Verlag in 2002. He has published 12 chapters in edited books and more than 170 papers in peer refereed conference proceedings. He has been supervising or co-supervising 12 Ph.D. and 18 Master students.