

## Chapter 2

# Introduction to ILC: Concepts, Schematics, and Implementation

**Abstract** In this chapter we review several important concepts, basic schematics, and implementation issues associated with real-time ILC. We first introduce five basic configurations of ILC for linear processes, including the previous cycle learning, current cycle learning, previous and current cycle learning, cascade learning, and incremental cascade learning. Next, we focus on ILC for non-linear processes, make clear two conditions intrinsic to ILC applications – the global Lipschitz continuity condition and identical initialization condition, and explore possible extensions. Finally we discuss three practical issues encountered in real-time ILC implementation – repetitive control tasks, robustness and filter design, as well as the sampling effect.

## 2.1 ILC for Linear Systems

In this section the concepts and schematics of ILC will be briefly reviewed for linear systems.

### 2.1.1 Why ILC?

Consider a control task that requires the perfect tracking of a pre-specified reference trajectory, for example moving and fixing parts in an assembly line, or temperature control of a batch reactor in the pharmaceutical industry. The common features of this class of control problems are 1) the task must be finished in a finite duration ranging from milliseconds to days, 2) the reference trajectory must be strictly followed from the very beginning of the execution, 3) the task is repeated from trial to trial, from batch to batch, from run to run, or in general from iteration to iteration, under the same conditions.

We face a new class of control tasks: perfect tracking in a finite interval under a repeatable control environment, where the repeatable control environment stands

for the repeatability of the process under the given control task that repeats and in the presence of repeated disturbances.

Most existing control methods including adaptive or robust control, may not be suitable for such a class of tasks because of two reasons. First, these control methods are characterized by the asymptotic convergence, thus it is difficult to guarantee a perfect tracking even if the initial discrepancy is zero. Second and more importantly, those control methods are not able to “learn” from previous task execution that may succeed or fail. Without learning, a control system can only produce the same performance without improvement even if the task is repeatedly executed. ILC was proposed to meet this kind of control requirement [4, 14, 83]. The idea of ILC is straightforward: using control information of the preceding execution to improve the present execution. This is realized through memory-based learning.

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

### 2.1.2 Previous Cycle Learning

The configuration of a previous cycle learning (PCL) scheme is shown in Fig. 2.1. Here, the subscript  $i$  denotes the  $i$ th iteration. Notations  $Y_{r,i}(s)$ ,  $Y_i(s)$ ,  $U_i(s)$  and  $E_i(s)$  denote the reference signals, output signals, control signals, and error signals, respectively at the  $i$ th iteration.  $P(s)$  and  $C_i(s)$  denote the transfer functions of the plant and the feedforward compensator, respectively. When the system performs the same control task,  $Y_{r,i+1}(s) = Y_{r,i}(s)$ . The MEM labelled with  $Y$ ,  $Y_r$  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).

Assume that the control task is repeated for all iterations, that is,  $Y_r = Y_{r,i} = Y_{r,i+1}$ . According to the PCL configuration shown in Fig. 2.1,

$$\begin{aligned} Y_i &= PU_i \\ E_i &= Y_r - Y_i \\ U_{i+1} &= U_i + C_i E_i. \end{aligned} \tag{2.1}$$

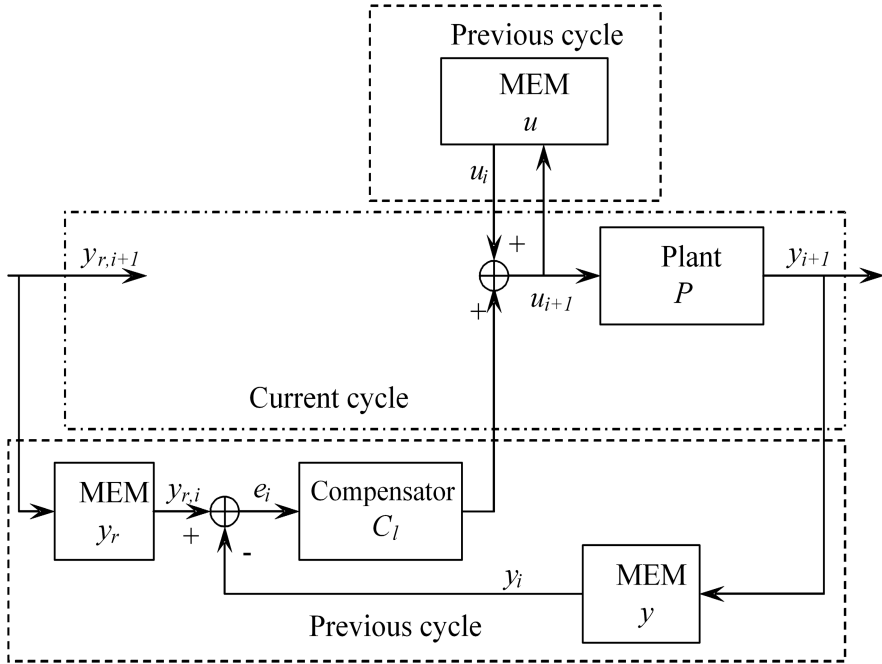
Equation 2.1 is the PCL updating law. It is called previous cycle learning simply because only the previous cycle control signals  $U_i$  and previous error signals  $E_i$  are used to form the current cycle control input  $U_{i+1}$ . It is an open-loop control in the time domain, but a closed-loop control in the iteration domain.

The learning convergence condition for PCL can be derived as,

$$\begin{aligned} E_{i+1} &= Y_r - Y_{i+1} \\ &= Y_r - PU_{i+1} \end{aligned}$$

$$\begin{aligned}
&= Y_r - P(U_i + C_l E_i) \\
&= Y_r - P U_i + P C_l E_i \\
&= (1 - P C_l) E_i \\
\Rightarrow \quad \frac{E_{i+1}}{E_i} &= 1 - P C_l \\
\Rightarrow \quad \left\| \frac{E_{i+1}}{E_i} \right\| &= \|1 - P C_l\| \leq \gamma < 1, \tag{2.2}
\end{aligned}$$

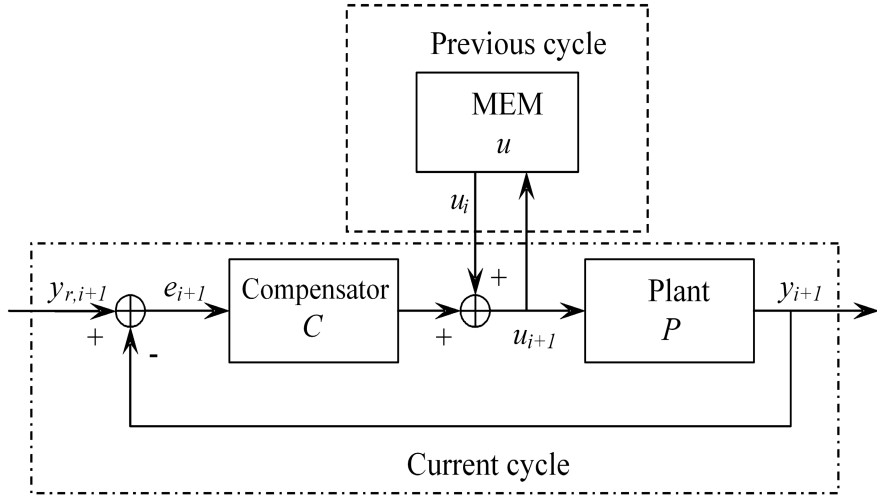
where the norm  $\|\cdot\|$  is the infinity norm for all frequencies  $\omega \in \Omega$ ,  $\Omega = [\omega_a, \omega_b]$ ,  $\omega_b > \omega_a \geq 0$ .  $\Omega$  denotes the frequency band of interest, or the frequency band that matches the bandwidth of a controller. Clearly, as far as the tracking error signals of the 1st iteration,  $E_0$ , is finite, then  $\|E_i\| \leq \gamma^i \|E_0\| \rightarrow 0$  as  $i \rightarrow \infty$ . It can also be seen that, for a process  $P$ , an appropriate choice of  $C_l$  will make a smaller  $\gamma$  and hence expedite the learning process.



**Fig. 2.1** The schematics of PCL

### 2.1.3 Current Cycle Learning

Due to the open-loop nature in the current cycle, PCL could be sensitive to small and non-repeatable perturbations. This can be improved by a feedback-based learning if the loop can be closed appropriately in the time domain, leading to the current cycle control (CCL). The configuration of the CCL scheme is shown in Fig. 2.2.



**Fig. 2.2** The schematics of CCL

Accordingly, the updating law of the CCL scheme is

$$U_{i+1} = U_i + CE_{i+1}, \quad (2.3)$$

where  $C$  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.

For the repeated control task  $Y_r$ , the convergence condition for CCL is derived as follows

$$\begin{aligned} E_{i+1} &= Y_r - Y_{i+1} \\ &= Y_r - PU_{i+1} \\ &= Y_r - P(U_i + CE_{i+1}) \\ &= Y_r - PU_i - PCE_{i+1} \\ (1 + PC)E_{i+1} &= E_i \\ \Rightarrow \frac{E_{i+1}}{E_i} &= \frac{1}{1 + PC} \end{aligned}$$

$$\Rightarrow \quad \left\| \frac{E_{i+1}}{E_i} \right\| = \left\| \frac{1}{1+PC} \right\| \leq \gamma < 1. \quad (2.4)$$

It can be seen that PCL and CCL are functioning in a complementary manner. PCL requires the factor  $\|1 - PC_l\|$  to be below 1 for all frequencies within the band  $\Omega$ , and often leads to a low gain  $C_l$ . CCL requires the factor  $\|1 + PC\|$  to be above 1 for all frequencies within the band  $\Omega$ , and often a high gain  $C$  is preferred as far as the closed-loop stability is guaranteed. Note that the memory arrays required for CCL are half of the PCL, because MEM for  $E_i$  is not required. Memory size is an important factor when implementing an ILC scheme.

### 2.1.4 Previous and Current Cycle Learning

Generally speaking, it may be difficult to find a suitable  $C_l$  such that (2.2) is satisfied, *i.e.*  $\|1 - PC_l\|$  is strictly less than 1 for all frequencies in  $\Omega$ . Likewise, it may be a difficult job to find a suitable  $C$  such that the denominator in (2.4) is strictly larger than 1 for all frequencies in  $\Omega$ . There is only 1 degree of freedom (DOF) in the controller design for both schemes. By pairing PCL and CCL together to form the previous and current cycle learning (PCCL), there is a possibility that the learning convergence will be improved. This can be achieved if the convergence conditions (2.2) and (2.4) can complement 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. 2.3.

For the repeated control task  $Y_r$ , it can be easily derived that the convergence condition is

$$\frac{\|1 - PC_l\|}{\|1 + PC\|},$$

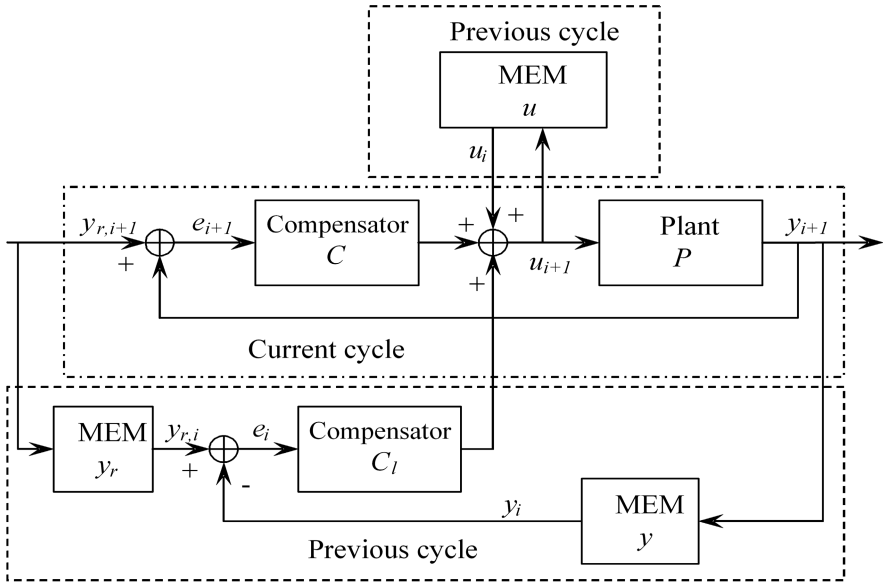
that is, the multiplication of both conditions of PCL and CCL. To demonstrate the advantage of the 2 DOF design with PCCL, an example is shown in Fig. 2.4, where

$$1 - PC_l = \frac{s^2 - 3}{3s^2 + 3s + 5}$$

and

$$1/(1 + PC) = \frac{s^2 + 20s + 80}{2s^2 + 10s + 130}.$$

Note that neither the PCL condition (2.2) nor the CCL condition (2.4) holds for all frequencies within the band  $\Omega = [10^{-2}, 10^2]$  rad/s, but the convergence condition with the integrated PCCL, which is the superposition of PCL and CCL in the Bode plot, does hold for all frequencies.



**Fig. 2.3** The schematics of a PCCL

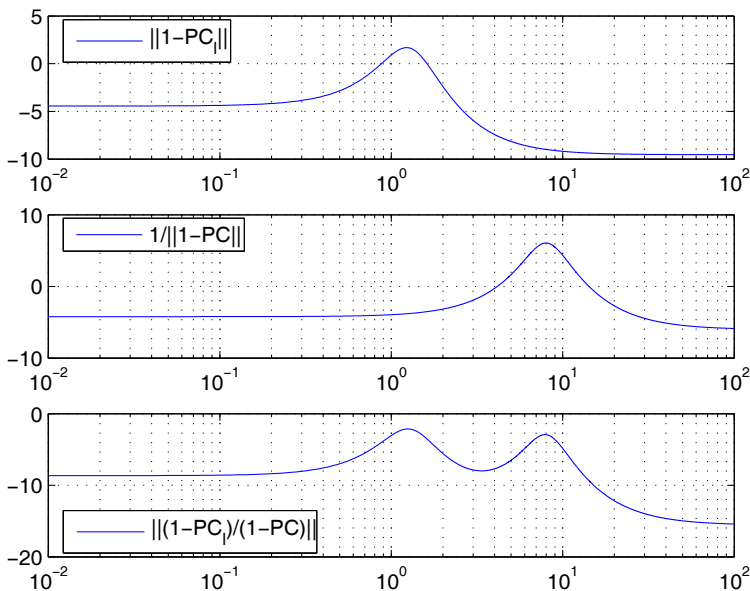
### 2.1.5 Cascade ILC

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 Figs. 2.1, 2.2 and 2.3, a new control block is embedded into the control loop. Such an embedded structure is the common structure for most existing real-time ILC schemes. Hence, when an ILC mechanism is to be incorporated into an existing control system, either the core execution program needs to be re-written 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 constraints. For instance, a rapid thermo-processing device in the wafer industry costs millions of dollars, and the only tunable part is a number of setpoints. In such circumstance, the cascade learning method is suitable as it modifies only the reference trajectory iteratively to improve the control performance.

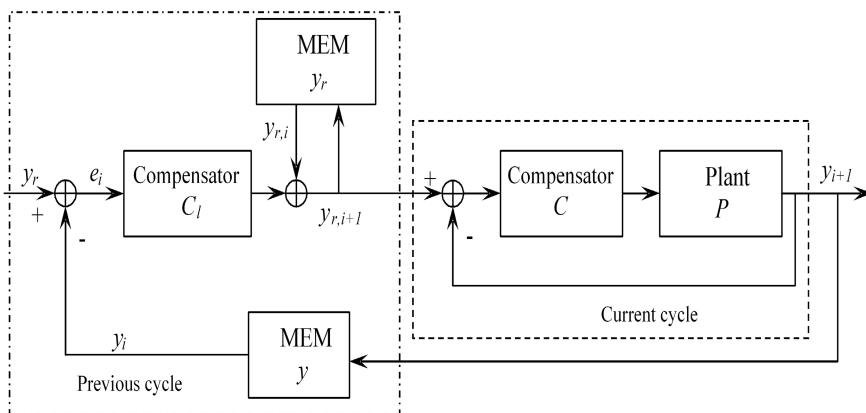
The schematics of such an ILC is demonstrated in Fig. 2.5. It can be seen that the ILC block is “cascaded” to the existing control loop or performed as an outer feedback loop in the iteration domain. The ILC with the cascade structure will use the modified reference signals and the actual system output of the previous cycle to generate the new reference signals for the current cycle. Owing to the cascade structure, the ILC need not be embedded into the existing control loop, thus avoiding any re-configuration of the system hardware or the core execution program. What

is needed is essentially some re-programming of reference signals, which can be easily carried out in real applications.

According to Fig. 2.5, we have



**Fig. 2.4** The complementary role of PCL and CCL



**Fig. 2.5** The schematics of a cascade ILC structure

$$\begin{aligned}
Y_i &= GY_{r,i} \\
E_i &= Y_r - Y_i \\
Y_{r,i+1} &= Y_{r,i} + C_l E_i \\
Y_{r,0} &= Y_r,
\end{aligned} \tag{2.5}$$

where  $G = PC/(1 + PC)$  denotes the closed-loop transfer function;  $Y_r$  is the original reference repeated over a fixed operation period;  $Y_{r,i}$  is the reference signal modified via learning for the inner current cycle control loop. According to the learning control law (2.5), the convergence condition for the cascade ILC can be derived as

$$\begin{aligned}
E_{i+1} &= Y_r - Y_{i+1} \\
&= Y_r - GY_{r,i+1} \\
&= Y_r - G(Y_{r,i} + C_l E_i) \\
&= Y_r - GY_{r,i} - GC_l E_i \\
&= (1 - GC_l)E_i \\
\Rightarrow \quad \frac{E_{i+1}}{E_i} &= 1 - GC_l \\
\Rightarrow \quad \left\| \frac{E_{i+1}}{E_i} \right\| &= \left\| 1 - \frac{PCC_l}{1 + PC} \right\| \leq \gamma < 1.
\end{aligned}$$

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

### 2.1.6 Incremental Cascade ILC

From the cascade ILC (2.5), we can derive the relationship between the original reference trajectory,  $Y_r$ , and the iteratively revised reference trajectory,  $Y_{r,i+1}$ , as below

$$\begin{aligned}
Y_{r,i+1} &= Y_{r,i} + C_l E_i \\
&= Y_{r,i-1} + \sum_{j=0}^1 C_l E_{i-j} \\
&= \dots = \\
&= Y_{r,0} + \sum_{j=0}^i C_l E_{i-j} \\
&= Y_r + \sum_{j=0}^i C_l E_{i-j}.
\end{aligned} \tag{2.6}$$

The original reference is modified with the errors  $\sum_{j=0}^i C_l E_{i-j}$ . Note that, if errors  $E_0, \dots, E_i$  are zero,  $Y_{r,i+1} = Y_r$ , that is, the original reference shall be used. In other words, if the current cycle feedback loop is able to achieve perfect tracking with



regard to the original reference  $Y_r$ , then the cascade ILC shall retain the original reference trajectory.

A potential problem associated with the cascade ILC in real-time applications is the filtering. In practical control processes with ILC, filters are widely applied and are imperative in general, due to the existence of non-repeatable perturbations or measurement noise. We will address the filtering issue in a later part of this chapter. A common practice in ILC applications is to design a low-pass filter or a band-pass filter,  $F[\cdot]$ , for the preceding control signals stored in memory, namely

$$U_{i+1} = F[U_i] + C_l E_i$$

for the embedded ILC, or

$$Y_{r,i+1} = F[Y_{r,i}] + C_l E_i$$

for the cascade ILC. The cascade ILC with filtering becomes

$$\begin{aligned} Y_{r,i+1} &= F[Y_{r,i}] + C_l E_i \\ &= F^2[Y_{r,i-1}] + \sum_{j=0}^1 F^j[C_l E_{i-j}] \\ &= \dots \\ &= F^i[Y_{r,0}] + \sum_{j=0}^i F^j[C_l E_{i-j}] \\ &= F^i[Y_r] + \sum_{j=0}^i F^j[C_l E_{i-j}], \end{aligned} \quad (2.7)$$

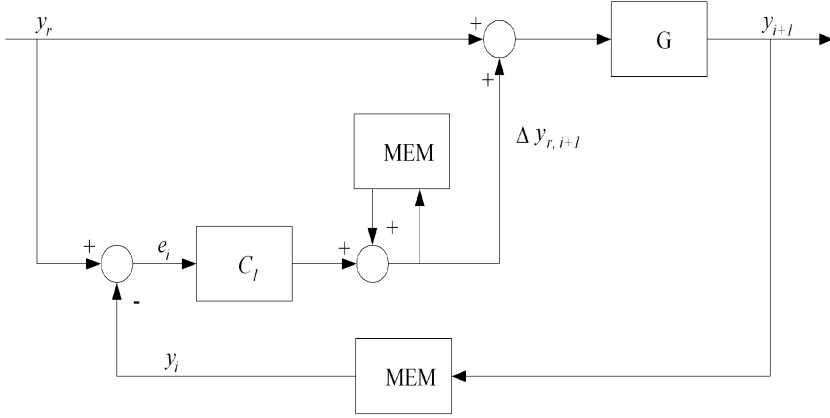
where  $F^0[x] = x$ . It is clear that, even if errors  $E_0, \dots, E_i$  are all zero,  $Y_{r,i+1} = F^i[Y_r] \neq Y_r$ , that is, the original reference has been changed. In other words, while the feedback loop is able to achieve perfect tracking with regard to the original reference  $Y_r$ , the cascade ILC is unable to retain the original reference trajectory, instead it is only able to provide a filtered one. As a result,  $Y_r$  will be distorted by  $F^i[\cdot]$  with additional magnitude change and phase lag, usually the larger the number of filtering times  $i$ , the larger the distortion. Clearly the original tracking task cannot be achieved in such a circumstance.

A solution to this problem is to introduce an incremental structure in the cascade ILC, as shown in Fig. 2.6. The learning control law is

$$\begin{aligned} Y_{r,i+1} &= Y_r + \Delta Y_{r,i+1} \\ \Delta Y_{r,i+1} &= \Delta Y_{r,i} + C_l E_i \\ \Delta Y_{r,0} &= 0. \end{aligned} \quad (2.8)$$

It is easy to derive that incremental learning is the same as the previous cascade learning (2.6),

$$Y_{r,i+1} = Y_r + \Delta Y_{r,i+1}$$



**Fig. 2.6** The schematics of an incremental cascade ILC structure

$$\begin{aligned}
 &= Y_r + \Delta Y_{r,i} + C_l E_i \\
 &= \dots \\
 &= Y_r + \sum_{j=0}^i C_l E_{i-j}.
 \end{aligned} \tag{2.9}$$

Now, adding a filter such that  $\Delta Y_{r,i+1} = F[\Delta Y_{r,i}] + C_l E_i$ , we can derive

$$\begin{aligned}
 Y_{r,i+1} &= Y_r + \Delta Y_{r,i+1} \\
 &= Y_r + F[\Delta Y_{r,i}] + C_l E_i \\
 &= Y_r + F^2[\Delta Y_{r,i-1}] + \sum_{j=0}^1 F^j[C_l E_{i-j}] \\
 &= \dots \\
 &= Y_r + \sum_{j=0}^i F^j[C_l E_{i-j}].
 \end{aligned} \tag{2.10}$$

It can be seen that the filtering applies to the past tracking errors. If errors  $E_0, \dots, E_i$  are all zero,  $Y_{r,i+1} = Y_r$ , that is, the original reference will not be changed.

## 2.2 ILC for Non-linear Systems

In practical applications, most plants are characterized by non-linear dynamics. In this section, we investigate the applicability of linear ILC schemes for non-linear dynamical processes. Consider the following non-linear plant

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), u(t)) \quad \mathbf{x}(0) = \mathbf{x}_0$$

$$y(t) = g(\mathbf{x}(t), u(t)), \quad (2.11)$$

where  $t \in [0, T]$ ,  $\mathbf{x}(t) \in \mathcal{R}^n$ ,  $y(t) \in \mathcal{R}$  and  $u \in \mathcal{R}$ ,  $\mathbf{f}(\cdot)$  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 [4] a typical yet the simplest first-order linear iterative learning control scheme is proposed

$$u_{i+1}(t) = u_i(t) + \beta e_i(t), \quad (2.12)$$

where  $e_i = y_r - y_i$ , and  $\beta$  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

$$\begin{aligned} |u_{i+1}|_\lambda &\leq |1 - \beta g_u| |u_i|_\lambda \\ |1 - \beta g_u| &= \gamma < 1, \end{aligned} \quad (2.13)$$

where  $g_u(\mathbf{x}, u, t) = \frac{\partial g}{\partial u}$ , and  $|\cdot|_\lambda$  is the time-weighted norm defined as  $\max_{t \in [0, T]} e^{-\lambda t} |\cdot|$ .

The learnability 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  $\beta$ , the convergence condition (2.13) can be fulfilled if the interval  $[\alpha_1, \alpha_2]$  is known *a priori*.

There are two conditions indispensable for linear ILC to be applicable to the non-linear process (2.11): the global Lipschitz continuity condition (GLC) and identical initialization condition (perfect resetting).

### 2.2.1 Global Lipschitz Continuity Condition

The GLC condition states that  $\mathbf{f}(\mathbf{x}, u, t)$  and  $g(\mathbf{x}, u, t)$  are global Lipschitz continuous with respect to the arguments. For  $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{R}^n$ , and  $u_1, u_2 \in \mathcal{R}$ , there exist constants  $L_f, L_g$  such that

$$\|\mathbf{f}(\mathbf{x}_1, u_1) - \mathbf{f}(\mathbf{x}_2, u_2)\| \leq L_f(\|\mathbf{x}_1 - \mathbf{x}_2\| + |u_1 - u_2|)$$

and

$$\|g(\mathbf{x}_1, u_1) - g(\mathbf{x}_2, u_2)\| \leq L_g(\|\mathbf{x}_1 - \mathbf{x}_2\| + |u_1 - u_2|).$$

Clearly, a linear plant is a special case of GLC functions. For instance,  $\mathbf{f} = \mathbf{A}\mathbf{x} + \mathbf{b}u$  has two Lipschitz constants denoted by an induced matrix norm  $\|\mathbf{A}\|$  and a vector norm  $\|\mathbf{b}\|$ .

In practical applications, it might not be easy to directly check whether a non-linear function  $\mathbf{f}(\mathbf{x})$  satisfies the GLC condition. A more convenient way is to derive the Jacobian of the non-linear function

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}}.$$

From the continuity of  $\mathbf{f}$ , applying the mean value theorem leads to

$$\mathbf{f}(\mathbf{x}_1) - \mathbf{f}(\mathbf{x}_2) = \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \Big|_{\mathbf{x}=\xi} (\mathbf{x}_1 - \mathbf{x}_2)$$

where  $\xi \in [\mathbf{x}_1, \mathbf{x}_2]$ . Hence we obtain a condition similar to GLC

$$\|\mathbf{f}(\mathbf{x}_1) - \mathbf{f}(\mathbf{x}_2)\| \leq \left\| \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \right\| \|\mathbf{x}_1 - \mathbf{x}_2\|$$

and the Lipschitz constant can be evaluated by

$$\max_{\xi \in [\mathbf{x}_1, \mathbf{x}_2]} \left\| \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \Big|_{\mathbf{x}=\xi} \right\|.$$

In the above derivation, the variables  $\xi$  are confined to a region specified by  $[\mathbf{x}_1, \mathbf{x}_2]$ . A sufficient condition for a non-linear function to be globally Lipschitz continuous is that its Jacobian is globally bounded, that is, uniformly bounded for any  $\xi \in \mathcal{R}^n$ . In fact, the quantity  $g_u$  in the learning convergence condition (2.13) is exactly the Jacobian.

Many engineering systems do not meet the GLC condition. For instance, as we will discuss later in this book, the current–torque (input–output) relationship of a switched reluctance motor contains a non-linear factor  $xe^{ax}$ , where  $x$  is the stator current and  $a$  is a bounded parameter. Its Jacobian is  $(1 + ax)e^{ax}$ , which is not uniformly bounded for any  $x \in \mathcal{R}$ . Nevertheless, the physical variables are often limited. For example, all electrical drives are equipped with current limiters to confine the maximum current to flow through the stator winding. As such, the Jacobian is always finite, subsequently the GLC condition is satisfied for the switched reluctant motor under any operating conditions.

In practical applications we also encounter such systems

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{f}(\mathbf{x}(t), u(t)) & \mathbf{x}(0) &= \mathbf{x}_0 \\ y(t) &= g(\mathbf{x}(t)), \end{aligned} \tag{2.14}$$

where the output  $y$  is not a function of the input  $u$ . In such circumstances, the preceding simple learning control law (2.12), usually called the P-type ILC law, does not work. Let us choose  $\dot{e}$  as the new output variable, then the new input–output relationship is

$$\dot{e}(t) = \frac{\partial g(\mathbf{x}(t))}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}(t), u(t)). \tag{2.15}$$

A D-type ILC law is constructed accordingly

$$u_{i+1}(t) = u_i(t) + \beta \dot{e}_i(t). \tag{2.16}$$

The convergence condition is determined by the relationship

$$\begin{aligned} |u_{i+1}|_{\lambda} &\leq |1 - \beta J| |u_i|_{\lambda} \\ |1 - \beta J| &= \gamma < 1, \end{aligned} \quad (2.17)$$

where

$$J = \left( \frac{\partial g}{\partial \mathbf{x}} \right)^T \frac{\partial \mathbf{f}}{\partial u}$$

is the Jacobian of the new input–output relationship.

Note that what is ensured by the convergence condition (2.17) is the asymptotic convergence of the error derivative  $\Delta \dot{y}_i(t) \rightarrow 0$  as  $i \rightarrow \infty$ . By virtue of the identical initial condition  $e_i(0)$ , as we will discuss next, the property  $\dot{e}_i(t) = 0$  implies  $e_i(t) = 0$  for any  $t \in [0, T]$ .

### 2.2.2 Identical Initialization Condition

Initial conditions, or initial resetting conditions, play a fundamental role in all kinds of iterative learning control methods [97]. ILC tries to make a “perfect” tracking in a finite time interval, whereas other control methods aim at an asymptotical convergence in an infinite time interval.

In practice, a control task must be finished in a finite time interval, for example the track following control of a hard disk drive with the duration in milliseconds, or temperature control of a batch reactor with the duration in hours or days. Note that these control problems require perfect tracking from the beginning. Consequently many existing control methods with the asymptotical convergence property are not suitable. A perfect tracking from the beginning demands the perfect initial condition, that is, the identical initialization condition (*i.i.c.*). In fact, in order to fulfill such a control task, other control methods such as adaptive control or robust control will also need the same condition.

Why do most existing control methodologies not require this condition? This is because they allow the control task to be achieved asymptotically in an infinite time interval, hence the initial discrepancy is not a problem. ILC cannot enjoy such an asymptotical convergence in the time domain because the task must be finished in a finite time interval, and requires perfect tracking over the entire transient period including the initial one.

In many practical control problems, the *i.i.c.* can be easily satisfied. For instance, most mechatronics systems have homing or re-setting mechanisms. Thus, we know where the process starts from. A batch reactor in a bio-chemical process always starts from a known room temperature that is under regulation.

There are three structural initialization conditions [153] used in ILC: two are state related and one is output related, mathematically expressed as (i)  $\mathbf{x}_i(0) = \mathbf{x}_0(0)$ , (ii)  $\mathbf{x}_i(0) = \mathbf{x}_r(0)$ , (iii)  $y_i(0) = y_r(0)$ . Note that the second condition implies the first condition, hence is more restrictive. The third condition is required in particular by

the D-type ILC as previously discussed. For strictly repeatable systems like (2.14), the second condition also implies the third condition.

In some practical applications, however, the perfect initial re-setting may not be obtainable. Five different initial conditions were summarized in [158],

- (a) identical initial condition;
- (b) progressive initial condition, *i.e.* the sequence of initial errors belong to  $l^2$ ;
- (c) fixed initial shift;
- (d) random initial condition within a bound;
- (e) alignment condition, *i.e.* the end state of the preceding iteration becomes the initial state of the current iteration.

The preceding three structural initialization conditions (i), (ii), (iii) were special cases of the condition (a). We face an important issue in ILC applications: the robustness under conditions (b), (c), (d) and (e). Most relevant studies done along this line were discussed in the literature survey of [158]. The simplest way is to revise the ILC updating law with a forgetting factor. For example, the learning law (2.12) becomes

$$u_{i+1}(t) = \alpha u_i(t) + \beta w_i(t), \quad (2.18)$$

where  $\alpha \in (0, 1]$  is a forgetting factor, and  $w_i(t)$  stands for one of the quantities  $e_i(t)$ ,  $e_{i+1}(t)$ , or  $\dot{e}_i(t)$ . The effect of  $\alpha$  is to prevent the influence of an initial discrepancy from going unbounded. Let us show this robustness property. Assume that the initial error,  $w_i(0)$ , is randomly distributed within a bound of  $\varepsilon$ , *i.e.* the condition (d) that is considered the most difficult yet most practical scenario. Substituting (2.18) to itself repeatedly with descending  $i$  we have

$$\begin{aligned} u_{i+1}(0) &= \alpha u_i(0) + \beta w_i(0) \\ &= \alpha^2 u_{i-1}(0) + \alpha \beta w_{i-1}(0) + \beta w_i(0) \\ &= \dots \\ &= \alpha^{i+1} u_0(0) + \beta \sum_{j=0}^i \alpha^{i-j} w_j(0). \end{aligned} \quad (2.19)$$

Initial control  $u_0(0)$  is always finite in practical problems, hence  $\alpha^{i+1} u_0(0) \rightarrow 0$  when  $i \rightarrow \infty$ . Therefore, as  $i \rightarrow \infty$ ,

$$|u_{i+1}(0)| \leq \beta \sum_{j=0}^i \alpha^{i-j} |w_j(0)| \quad (2.20)$$

$$\leq \beta \varepsilon \frac{1 - \alpha^{i+1}}{1 - \alpha}, \quad (2.21)$$

which is obviously finite as far as  $\alpha \in (0, 1)$ . In real-time ILC implementations, some prior knowledge is needed in choosing an appropriate value for  $\alpha$ . A recommended range is  $[0.95, 0.99]$ . However, from (2.21) the error bound could be 20–100 times higher than  $\varepsilon$  when choosing  $\alpha \in [0.95, 0.99]$ . A lower  $\alpha$  can reduce

the error bound, but meanwhile the learning effect is inevitably weakened. To avoid such a tradeoff, we need to look into initial conditions (b), (c), (d) in more detail.

The issue concerned with the condition (b) is whether the accumulated influence of the initial discrepancy will vanish if the initial error is itself vanishing gradually, *i.e.* under the progressive condition (b). Assume that  $w_i(0)$  is gradually vanishing with a geometric rate, hence its upper bound can be expressed by  $\gamma_1^i \varepsilon$  with  $\gamma_1 \in (0, 1)$ . Denote  $\gamma_2 = \max\{\alpha, \gamma_1\}$ . Clearly  $\gamma_2 \in (0, 1)$ . According to the preceding derivation,

$$\begin{aligned} |u_{i+1}(0)| &\leq \beta \sum_{j=0}^i \alpha^{i-j} \gamma_1^j \varepsilon \\ &\leq \beta \varepsilon \sum_{j=0}^i \gamma_2^{i-j} \gamma_2^j \\ &= (i+1) \gamma_2^i \beta \varepsilon \end{aligned} \quad (2.22)$$

which goes to zero when  $i \rightarrow \infty$ . In such circumstances, we can choose  $\alpha$  sufficiently close to 1 without sacrificing the ultimate error bound.

The condition (c) indicates the presence of a fixed initial shift, for example an XY table homing position is not the starting point of the task, or the room temperature is not the starting temperature of a reactor. A set of initial state controllers, such as proportional integral (PI) controllers, can be used to bring the initial states to the desired values, then the tracking process starts. This way is practically easy and effective, as it is a point-to-point or setpoint control problem that is much easier than tracking control problems. If the initial state control is not possible, an alternative way is to rectify the target trajectory at the initial phase. For D-type ILC, we can set  $\mathbf{y}_r(0) = \mathbf{y}_i(0)$ , because the output is accessible [123]. On the other hand, *i.e.* is still open when the initial states  $\mathbf{x}_i(0)$  are not accessible.

The scenario (d) is most frequently encountered in practice. Let us consider  $w_i(0)$  to be a zero-mean white noise along the iteration axis  $i$  and a standard deviation  $\sigma < \infty$ . The sum on the right-hand side of (2.19)

$$\sum_{j=0}^i \alpha^{i-j} w_j(0)$$

can be viewed as the output of a first-order linear system

$$\frac{1}{z - \alpha}$$

driven by a random input  $w_i(0)$ . The power spectrum density of the sum is

$$\frac{\sigma^2}{(z - \alpha)^2}$$

which has a finite mean square value  $\sigma^2/(1-\alpha)^2$ . Therefore, the power spectrum density of  $u_i(0)$  is proportional to  $\sigma$ , and bounded owing to the introduction of the forgetting factor  $\alpha$ .

In real-time applications to motion systems, the zero initial condition can be easily guaranteed for higher-order states such as velocity, acceleration, as far as motion systems come to a full stop after every iteration. Thus, only the position state needs a perfect re-set, which can be easily achieved by means of a PI controller with a sufficiently long settling time.

## 2.3 Implementation Issues

### 2.3.1 Repetitive Control Tasks

ILC is essentially for tracking in a finite time interval, which is different from repetitive tasks – either tracking a periodic trajectory or rejecting a periodic exogenous disturbance. Repetitive control tasks are continuous in the time domain, such as the speed control of an electrical drive, or voltage control of a power supply. The common feature of this class of control problems is the presence of some undesirable cyclic behavior when the system works around an operating point, for instance a power supply generates a DC voltage, but perturbed by 50 Hz harmonics. Repetitive control was developed to deal with this class of control tasks, but encounters difficulty when a non-linear process is concerned. In fact, we have shown experimentally that a simple repetitive control law below can effectively deal with non-linear dynamics

$$u(t) = u(t - T) + \beta e(t) \quad t \in [0, \infty), \quad (2.23)$$

where  $T$  is the period of the cyclic reference trajectory or cyclic disturbance. Unfortunately, we lack an effective analysis tool for repetitive control, as most repetitive control problems can only be analyzed in the frequency domain, despite the fact that the controller does perform well for non-linear processes.

Recently, three learning control methods were proposed for non-linear systems that perform repetitive control tasks. The first method uses the alignment condition (e) and employs Lyapunov methods in the analysis. It requires the plant to be parameterized, and all states to be accessible. This method is named as adaptive ILC [35, 37, 131, 152]. The second method extends (2.23) to the non-linear repetitive learning control laws in which periodic learning and robust control are integrated [30, 160]. The convergence analysis is conducted using the Lyapunov–Krasovskii functional. The third method is also a non-linear learning control method that stabilizes the process by a non-linear feedback controller, and meanwhile introduces a periodic signal [64]. Then, from [162] it is known that the closed loop will enter a cyclic steady state after a sufficiently long interval. Next, by changing the periodic signal, the system will enter another cyclic steady state after a sufficiently long



interval. The control system will converge asymptotically if the periodic signal is updated, once after the long interval, according to the learning law (2.23).

According to the third method, the repetitive control process can be converted into the repeated control process, subsequently the ILC analysis can be extended to repetitive control problems for a large class of non-linear processes. Note that the process is stable under feedback control but the undesirable cyclic behavior cannot be eliminated. Any change of the control signal will yield a new cyclic steady state. Thus, we can segment the horizon  $[0, \infty)$  into a sequence of finite intervals  $[iT, (i+1)T)$ ,  $i = 0, 1, \dots$ . Let  $\tau \in [0, T)$ . Then, for any instant  $t \in [0, \infty)$ , there exists an interval  $[iT, (i+1)T)$ , such that  $t = \tau + iT$ . Denote  $u_i(\tau) = u(\tau + iT)$ , then  $u(t - T) = u_{i-1}(\tau)$ . By segmenting  $\Delta y(t)$  in the same way as  $u(t)$ , the repetitive control law (2.23) can now be re-written as

$$u_{i+1}(\tau) = u_i(\tau) + \beta e_{i+1}(\tau) \quad \tau \in [0, T). \quad (2.24)$$

Comparing with the current cycle ILC law (2.3), it can be seen that both share the same form. Next,  $T$  is sufficiently long so that the closed-loop response will enter the cyclic steady state at the end of each interval. Learning control can be updated between any two consecutive intervals. A remaining problem is how to choose the period  $T$ . In several learning applications,  $T$  was chosen to be the lowest common multiple among cyclic factors that should be rejected. A concern is that the system may not reach its cyclic steady state within a short  $T$ .

### 2.3.2 Robustness and Filter Design

In practical applications, the textbook-like ILC algorithms, such as (2.1) and (2.3), may not function properly. Often, we can observe such a V-phenomenon (drop-rise phenomenon) : the tracking error decreases at the first few iterations, but increases when the iterative learning process continues. A common explanation for such a V-phenomenon is the presence of non-repeatable factors in the learning process. Indeed, none of the practical processes are perfectly repeatable. The system response always contains both the repeatable and non-repeatable signals. Learning of non-repeatable signals could be dangerous, because ILC is essentially a point-wise integration along the iteration axis. However, this explanation is not conclusive. Let us see what kind of non-repeatable disturbances will result in an increasing learning error sequence. Consider the process  $P(s)$  subject to an exogenous disturbance or measurement noise  $D_i(s)$

$$Y_i = PU_i + D_i.$$

When the ILC law (2.1) is used, the ILC tracking error is

$$\begin{aligned} E_{i+1} &= Y_r - Y_{i+1} \\ &= Y_r - PU_{i+1} - D_{i+1} \end{aligned}$$

$$\begin{aligned}
&= Y_r - P(U_i + C_l E_i) - D_{i+1} \\
&= (1 - PC_l)E_i - (D_{i+1} - D_i).
\end{aligned} \tag{2.25}$$

From (2.25), the repeatable perturbation  $D_{i+1} = D_i$  can be completely rejected. It was shown in the PCL discussion that, in the absence of the perturbations  $D_{i+1} - D_i$ , the convergence is determined by the factor

$$\|1 - PC_l\| \leq \gamma < 1$$

for frequencies of interest  $\omega \in \Omega$ ,  $\Omega = [\omega_a, \omega_b]$ . Note that ILC is unable to reject non-repeatable perturbations. This is also true for any ILC method. Our concern is, in what circumstances will ILC lead to a divergent control sequence when non-repeatable  $D_i$  is present.

In (2.25), denote  $\Delta_i = D_{i+1} - D_i$ , we have

$$E_{i+1} = (1 - PC_l)^{i+1} E_0 + \sum_{j=0}^i (1 - PC_l)^{i-j} \Delta_j. \tag{2.26}$$

The  $V$  learning phenomenon can be partially explained from the relationship (2.26). The tracking error reduction at the first few iterations is owing to the attenuation of the initial error  $(1 - PC_l)^{i+1} E_0$ , whereas the subsequent increment of the error is due to the accumulation of  $(1 - PC_l)^{i-j} \Delta_j$ . Note that the summation term in (2.26) may remain bounded when the magnitude of  $1 - PC_l$  is strictly below 1 for all frequencies. However, if the relative degree of  $PC_l$  is above 0, the magnitude of  $1 - PC_l$  approaches 1 when the frequency  $\omega$  is sufficiently high, leading to the occurrence of large tracking errors at high frequencies. The accumulation of  $\Delta_i$  is because of the integral action in the ILC laws (2.1) and (2.3).

To prevent such an undesirable “integration” for non-repeatable components, filtering technology has been widely used in practice. Two commonly adopted modifications for ILC robustification are the forgetting factor and the low-pass filtering. Consider the PCL scheme with the forgetting factor  $\alpha \in (0, 1]$

$$U_{i+1} = \alpha U_i + C_l E_i. \tag{2.27}$$

The tracking error is

$$\begin{aligned}
E_{i+1} &= Y_r - Y_{i+1} \\
&= Y_r - P U_{i+1} - D_{i+1} \\
&= Y_r - P(\alpha U_i + C_l E_i) - D_{i+1} \\
&= (\alpha - PC_l)E_i - [D_{i+1} - \alpha D_i - (1 - \alpha)Y_r].
\end{aligned} \tag{2.28}$$

Now, the convergence is determined by the factor  $\alpha - PC_l$ . When the relative degree of  $PC_l$  is above 0, the magnitude of  $\alpha - PC_l$  approaches  $\alpha < 1$ . Thus, the forgetting factor can effectively prevent the high-frequency perturbation from accumulating. The main drawback of the forgetting factor is the non-discriminant attenuation to all

frequencies including useful frequencies. To avoid the drawback of the forgetting factor, the constant forgetting factor  $\alpha$  can be replaced by a frequency-dependent low-pass filter (LPF)  $F(s)$

$$U_{i+1} = FU_i + C_l E_i, \quad (2.29)$$

where  $F$  has a cutoff frequency  $\omega_c$ .

At the low-frequency band  $\omega < \omega_c$ ,  $F$  is approximately equal to 1. The LPF modification works in the circumstances where  $D_i$  is negligible at the low-frequency band, and any attenuation of the reference trajectory  $Y_r$  should be avoided. At the high-frequency band  $\omega > \omega_c$ , the magnitude of  $F$  is less than 1, thus  $F$  works as a forgetting factor. In principle, if the power spectra of  $Y_r$  and  $D_i$  can be separated by  $\omega_c$ , the LPF modification yields satisfactory responses. If the power spectra of  $Y_r$  and  $D_i$  overlap for frequencies  $\omega > \omega_c$ , perfect tracking of  $Y_r$  is not achievable and the introduction of LPF prevents the worst case of divergence. If the perturbation  $D_i$  contains low-frequency or DC components, it is not advisable to further lower the cutoff  $\omega_c$ . Instead, the LPF should be revised such that the magnitude of  $F$  at the low-frequency band is strictly less than 1. If there is no prior knowledge about the power spectrum of  $D_i$ , the magnitude of the LPF should be chosen uniformly below 1, especially  $|F(0)| < 1$ . As a rule of thumb,  $|F(0)| \in [0.9, 0.99]$  is recommended.

Finally, we discuss the structure of the LPF. Among numerous filters, Butterworth LPF is most widely used owing to the smooth frequency response, as shown in Fig. 2.7, and the adjustable stiffness in the transition band by the order of the filter. The frequency response function of the Butterworth LPF has the following form

$$|F(j\omega)|^2 = \frac{1}{1 + (j\omega/j\omega_c)^{2N}}, \quad (2.30)$$

where  $N$  is the order of the filter. From Fig. 2.7, it can be seen that Butterworth LPF has the pass band at  $\omega = 0$  and the stop band at  $\omega \rightarrow \infty$ . At the cutoff frequency,  $|F(j\omega_c)| = 1/\sqrt{2}$ , regardless of the order of the filter. As the filter order increases, the transition from the pass band to the stop band gets steeper, namely approaching the ideal cutoff. In practical applications, the second-order Butterworth filter is often adequate to attenuate the high frequencies. Choosing the order  $N > 2$  though can further improve the stop-band property, the phase lag will also increase and finally lead to a poor learning result.

### 2.3.3 Sampled-data ILC

ILC are implemented using microprocessors, not only because of the recent advances in microprocessor technology, but also because of the nature of ILC – use of a memory array to record the past control information. The sampled-data ILC shows unique characteristics, and attracted a number of researchers to explore and develop

suitable ILC algorithms. Here, we briefly investigate three effects associated with the sampling.

The first and best known effect of sampling is the one-step sampling delay. In contrast to other control problems, ILC requires the identical initialization condition when aiming at the perfect tracking. Fortunately, PCL uses the past tracking error signals, hence the one-step delay can be easily compensated and the ILC law in the time domain is

$$u_{i+1}(k) = u_i(k) + \beta e_i(k+1), \quad (2.31)$$

where  $k$  is the sampling instance,  $\beta$  is a constant learning gain. Clearly, this one-step advance compensation cannot be implemented in CCL algorithms. Therefore in most ILC applications, a feedback  $C$  is first incorporated and the closed-loop transfer function is

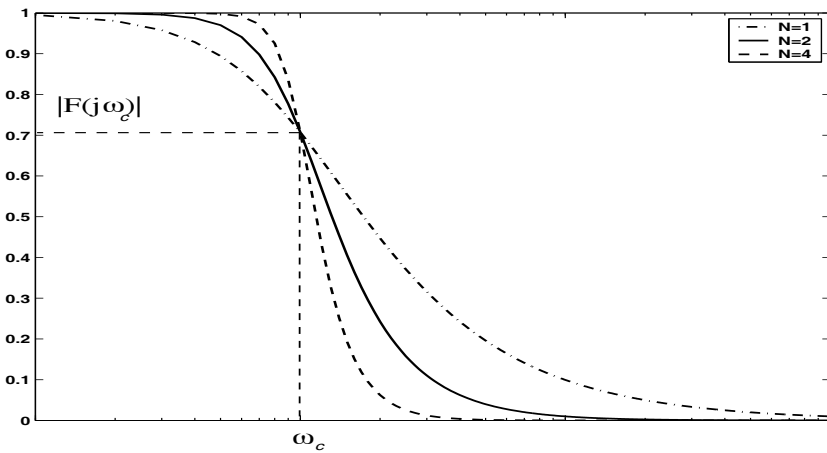
$$G = \frac{PC}{1+PC},$$

where  $P$  and  $C$  are transfer functions in the  $z$ -domain. Next, the ILC is designed for the closed-loop system, accordingly, the learning convergence condition is

$$\|1 - GC_l\| \leq \gamma < 1.$$

In a sense, this design is a PCCL, where the feedback  $C$  guarantees the closed-loop stability along the time axis, and the compensator  $C_l$  ensures the learning convergence along the iteration axis.

The second effect of sampling is related to the size of the sampling interval  $T_s$ , which directly determines the sampling frequency. From the sampling theorem, a smaller sampling interval is preferred and the sampling interval is a major factor that decides the tracking accuracy. However, a smaller sampling interval may degrade



**Fig. 2.7** Frequency response of Butterworth low-pass filters

the transient performance of ILC in the iteration domain. Consider a continuous-time process

$$\dot{\mathbf{x}} = A\mathbf{x} + B\mathbf{u}.$$

After sampling, the process is

$$\begin{aligned}\mathbf{x}(k+1) &= \Phi\mathbf{x}(k) + \Gamma\mathbf{u}(k), \\ \Phi &= e^{A\Delta} \quad \Gamma = \int_0^\Delta e^{A\tau} d\tau.\end{aligned}$$

When  $A$  is asymptotically stable, all the eigenvalues of  $\Phi$  are within the unit circle. The larger the  $T_s$ , the closer are eigenvalues of  $\Phi$  to the origin. Recent research [1] shows that a monotonic convergence along the iteration axis can be achieved if  $\|\Phi\|$  is sufficiently small. Therefore, a smaller sampling interval can improve the tracking accuracy in the time domain, whereas a larger sampling interval could improve the transient response of ILC in the iteration domain. A multi-rate ILC controller can be considered. For the closed-loop feedback the minimum available sampling interval should be used. For the learning loop we can first use a larger sampling interval to avoid a poor transient response, and reduce the sampling interval when the tracking error does not decrease further as the iteration proceeds.

The third effect of the sampling is associated with the relative degree of the original continuous-time process. It is known that sampling with zero-order hold can reduce the relative degree from  $n > 1$  in continuous time to 1 in the sampled data. As a consequence, it is adequate to use the ILC law (2.31) even if the original continuous process has a relative degree above 1. This property is highly desirable, as only one-step advance compensation is required. In comparison, if the ILC is designed in continuous time,  $n$ th-order differentiation is necessary, but the implementation of such a high-order differentiator is not an easy task, especially when measurement noise is present. Another effect, a side effect, from using a smaller sampling interval is the generation of a non-minimum phase model. It has been shown in [10], that a non-minimum phase discrete-time process will be generated if the original continuous-time process has a relative degree above 2 and the sampling interval is sufficiently small. As a result, perfect tracking is not possible. In such circumstances, multi-rate CCL is an appropriate choice, and the sampling interval for the learning updating mechanism should be chosen as long as possible as long as the tracking accuracy is satisfied.

## 2.4 Conclusion

This chapter focused on several important issues linked to the basic ILC configurations, GLC conditions, identical initialization conditions, filtering and sampling. There are many issues involved in the ILC implementation and they could not be covered in this chapter. Some of these issues will be further addressed in subse-

quent chapters when the real-time implementation problems are explored. Some other issues were explored in recent ILC literature, and are surveyed in the preceding chapter. No doubt many are still open problems and wait for us to solve in the future.



<http://www.springer.com/978-1-84882-174-3>

Real-time Iterative Learning Control  
Design and Applications

Xu, J.-X.; Panda, S.K.; Lee, T.H.

2009, XVI, 194 p., Hardcover

ISBN: 978-1-84882-174-3