# 1

# Introduction

## 1.1 General Overview of the Monograph

This monograph focuses on robust monotonically-convergent iterative learning control (MC-ILC) systems and stochastic iterative learning control systems. For robust MC-ILC design, parametric interval uncertainty models are considered. For stochastic ILC design, norm-bounded uncertainty, external disturbances, stochastic measurement noise, and intermittent measurements are considered, all in the iteration domain.

The monograph is organized into three parts and an appendix section. In the first part of the monograph, the ILC problem is described and motivated in Chapter 1. Then Chapter 2 gives an overview of the ILC literature, covering specifically the literature published between 1998 and 2004. In Chapter 3 the super-vector ILC (SVILC) framework is introduced for use in Chapters 4 to 8. The second part of the monograph considers interval ILC analysis (Chapter 4) and interval ILC synthesis (Chapter 5 and Chapter 6). The focus in the second part of the monograph is on plants with parametric interval uncertainty models. In addition to analysis and synthesis for such systems, it is shown how to develop suitable Markov interval models from state-space interval models. The third part of this monograph discusses $H_\infty$ SVILC design (Chapter 7) and stochastic ILC (Chapter 8). The focus in the third part of the monograph is on asymptotic stability and monotonic convergence conditions of ILC systems under assumptions of iteration-domain model uncertainty and stochastic disturbances and noise. In the Appendix section, a taxonomy of the ILC literature is presented and three fundamental interval computational problems are introduced and solved. Although these interval problems were initially motivated for solving interval ILC problems, due to their own completeness and their potential impact on robust control research in general, these results are carefully described in the appendices.

## 1.2 Iterative Learning Control

### 1.2.1 What Is Iterative Learning Control?

Control systems have played an important role in the development and advancement of modern civilization and technology. Control problems arise in practically all engineering areas and have been studied both by engineers and by mathematicians. Industrially, control systems are found in numerous applications: quality control in manufacturing systems, automation, network systems, machine tool control, space engineering, military systems, computer science, transportation systems, robotics, social systems, economic systems, biological/medical engineering, and many others. Mathematically, control engineering includes modeling, analysis, and design aspects. The key feature of control engineering is the use of feedback for performance improvement of a controlled dynamic system. The branches of modern control theories are broad and include classical control, robust control, adaptive control, optimal control, nonlinear control, neural networks, fuzzy logic, and intelligent control, with each branch being distinguished from the others based on the assumptions made about the properties of the systems to be controlled and the performance objectives of the specific methodology under consideration.

Iterative Learning Control (ILC) is one of the more recent control theories. ILC, which can be categorized as an intelligent control methodology,[1] is an approach for improving the transient performance of systems that operate repetitively over a fixed time interval. Although control theory provides numerous tools for improving the performance of a dynamic system, it is not always possible to achieve a desired level of performance. This may be due to the presence of unmodeled dynamics, parametric uncertainties, or disturbances and measurement noise exhibited during actual system operation. The inability to achieve a desired performance may also be due to the lack of suitable design techniques [298]. In particular, when the system is nonlinear, it is not easy to achieve perfect tracking using traditional control theory. However, for a specific class of systems – those that operate repetitively – ILC is a design tool that can help overcome the shortcomings of traditional controllers, making it possible to achieve perfect tracking or performance when there is model uncertainty or when we have a "blind" system.[2]

---

[1] From "Defining Intelligent Control – Report of the Task Force on Intelligent Control," IEEE Control Systems Society, Panos Antsaklis, Chair, Dec., 1993: "Intelligent control uses conventional control methods to solve lower level control problems ... conventional control is included in the area of intelligent control. Intelligent control attempts to build upon and enhance the conventional control methodologies to solve new, challenging control problems."

[2] "Blind" means we have little or no information about the system structure and its nonlinearities. We can only measure input/output signals such as voltage, velocity, position, etc.

Various definitions of ILC have been given in the literature. Some of them are quoted here.

- "The learning control concept stands for the repeatability of operating a given objective system and the possibility of improving the control input on the basis of previous actual operation data." Arimoto, Kawamura, and Miyazaki [29].
- ILC is a "... recursive online control method that relies on less calculation and requires less a priori knowledge about the system dynamics. The idea is to apply a simple algorithm repetitively to an unknown plant, until perfect tracking is achieved." Bien and Huh [43].
- "Iterative learning control is an approach to improving the transient response performance of a system that operates repetitively over a fixed time interval." Moore [298].
- "Iterative learning control considers systems that repetitively perform the same task with a view to sequentially improving accuracy." Amann, Owens, and Rogers [10].
- "ILC is to utilize the system repetitions as experience to improve the system control performance even under incomplete knowledge of the system to be controlled." Chen and Wen [66].
- ILC is a "... controller that learns to produce zero tracking error during repetitions of a command, or learns to eliminate the effects of a repeating disturbance on a control system output." Phan, Longman, and Moore [363].
- "The main idea behind ILC is to iteratively find an input sequence such that the output of the system is as close as possible to a desired output. Although ILC is directly associated with control, it is important to note that the end result is that the system has been inverted." Markusson [287].

All definitions about ILC will have their own emphases. However, a common feature of the definitions above is the idea of "repetition." Learning through a *predetermined* hardware repetition is the key idea of ILC. Hardware repetition can be thought of as a physical layer on the uniformly distributed time axis for providing experience to the mental layer of ILC. "Predetermined" means that the ILC system requires some postulates that define the learning environment of a control algorithm. A person learns about his/her living environment by experience, where the physical layer is the daily activity and the mental layer is the memory of strongly perceived events that are closely related with his/her interest. These strongly perceived events of the past provide knowledge to the human being that can be used to inform the person's current activity. In ILC, the current activity is a control force and the past experience is stored data. A difference between human learning and machine learning is in "predetermined." For a human being, knowledge by learning could be based on similarity and impression, whereas in a machine the initial setup, fixed time point, uniform sampling, repetitive desired trajectory, etc. are predetermined, which then determines the future of the

hardware machine. Thus, ILC is concerned with the problem of refining the input to a system that operates repetitively, so that the future behavior of the system is predetermined to improve its current operation over its past operation through the use of past experience.

Consider a system in an initial state to which a fixed-length input signal is applied. After the complete input has been applied, the system is returned to its initial state and the output trajectory that resulted from the applied input is compared to a desired trajectory. The error is used to construct a new input signal (of the same length) to be applied the next time the system operates. This process is then repeated. The goal of the ILC algorithm is to properly refine the input sequence from one trial to the next trial so that as more and more trials are executed the output will approach the desired trajectory.

The basic idea of ILC is illustrated in Figure 1.1. Standard assumptions are that the plant has stable dynamics or satisfies some kind of Lipschitz condition, that the system returns to the same initial conditions at the start of each trial, that the trial lasts for a fixed time $T_f$, and that each trial has the same length. A typical ILC algorithm for the architecture depicted in Figure 1.1 has the form

$$u_{k+1}(t) = u_k(t) + \gamma \frac{d}{dt} e_k(t), \tag{1.1}$$

where $u_k(t)$ is the system input and $e_k(t) = y_d(t) - y_k(t)$ is the error on trial $k$, with $y_k(t)$ the system output and $y_d(t)$ the desired response. For a large class of systems this algorithm can be shown to converge in the sense that as $k \to \infty$ we have $y_k(t) \to y_d(t)$ for all $t \in [0, T_f]$. Notice that this algorithm is noncausal. To see this more clearly, note that a discrete-time version of (1.1) can be given as

$$u_{k+1}(t) = u_k(t) + \gamma e_k(t+1), \tag{1.2}$$

where now $t$ is an integer. Clearly (1.2) is noncausal with respect to time. This is a key feature of ILC. Though the algorithm actually acts only on past data, the fact that the initial conditions are reset at the beginning of each trial allows us to do "noncausal" processing on the errors from the previous trial.

Based on the ILC system definition as depicted in Figure 1.1 and following the definitions quoted above, we will propose the following definition:

> ILC is an approach to improve the transient response performance of an unknown or uncertain hardware system that operates repetitively over a fixed time interval by eliminating the effects of a repeating disturbance and by using the previous actual operation data.

Finally, having defined ILC, it is important to point out the focus of ILC research, as clearly defined in the following quote:

- "We learned that ILC is about enhancing a system's performance by means of repetition, but we did not learn how it is done. This brings us to the
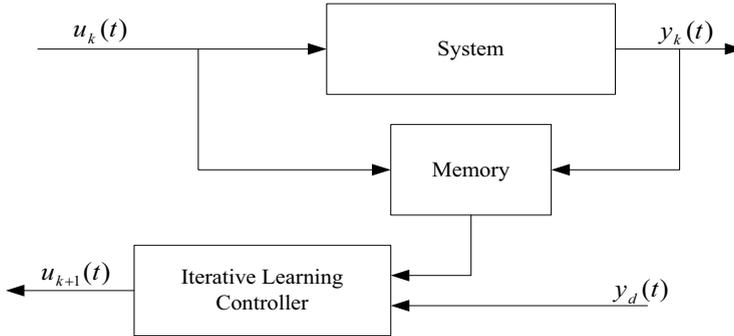
**Fig. 1.1.** Basic idea of ILC

core activity in ILC research, which is the construction and subsequent analysis of algorithms." Verwoerd [467].

Thus, the key question of ILC is how to eliminate the periodic disturbance and how to use the past information for the current trial. As we will see below, if the system uncertainty and external disturbances are predetermined on the uniformly distributed repetitive time axis, all of these effects, including the actual plant, could be considered as a predetermined model, so that finding an inverse of the deterministic system will be the main objective of ILC.

### 1.2.2 Classical ILC Update Law

As shown from the taxonomy given in Appendix A, the scope of ILC research is so wide that it is nearly impossible to describe all the branches of ILC. Thus, in this subsection we review only the basic ideas of classical ILC algorithms. Let us consider the following simple SISO linear repetitive system, in continuous time:

$$\dot{x}_k(t) = Ax_k(t) + Bu_k(t) \tag{1.3}$$
$$y_k(t) = Cx_k(t). \tag{1.4}$$

The control task is to servo the output $y_k(t)$ to track the desired output $y_d(t)$ for all time $t$ as the iteration $k$ increases. In classical ILC, the following basic postulates are required, although in recent ILC research, algorithms are sought so that these postulations could be somewhat broken or relaxed (adopted from page 2 of [66]):

- Every trial (pass, cycle, batch, iteration, repetition) ends in a fixed time of duration.
- Repetition of the initial setting is satisfied. That is, the initial state $x_k(0)$ of the objective system can be set to the same point at the beginning of each iteration.

- Invariance of the system dynamics is ensured throughout the repetition.
- The output $y_k(t)$ is measured in a deterministic way.

To give a flavor of ILC results, consider the learning control algorithm proposed in 1984 by Arimoto (and hence, called an "Arimoto-type" ILC law) [27, 28]:

$$u_{k+1}(t) = u_k(t) + \Gamma \dot{e}_k(t). \tag{1.5}$$

For this algorithm and the plant in (1.3–1.4), convergence is assured if

$$\|I - CB\Gamma\|_i < 1. \tag{1.6}$$

Arimoto also considered more general "PID-type" ILC algorithms of the form:

$$u_{k+1} = u_k + \Phi e_k + \Gamma \dot{e}_k + \Psi \int e_k dt. \tag{1.7}$$

In other types of algorithms, researchers have used gradient methods to optimize the gain $G_k$ in:

$$u_{k+1}(t) = u_k(t) + G_k e_k(t+1). \tag{1.8}$$

For ILC design purposes it is sometimes useful to specify the learning control algorithm in the frequency domain, for example:

$$U_{k+1}(s) = L(s)[U_k(s) + aE_k(s)]. \tag{1.9}$$

Note that in this last case the ILC gain is actually implied to be a linear time-invariant filter.

Many schemes in the literature can be classified with one of the algorithms given above. As such, it is possible to generalize the analysis by introducing an operator-theoretic notation. Let $T(\cdot)$ denote a general operator mapping an input space to an output space. Then the following theorem summarizes a number of technical convergence results for first-order[3] ILC systems:

**Theorem 1.1.** [298] *For the plant $y_k = T_s u_k$, the linear time-invariant learning control algorithm*

$$u_{k+1} = T_u u_k + T_e(y_d - y_k) \tag{1.10}$$

*converges to a fixed point $u^*(t)$ given by*

$$u^*(t) = (I - T_u + T_e T_s)^{-1} T_e y_d(t) \tag{1.11}$$

*with a final error*

---

[3] First-order ILC means that only data from the most recent (previous) trial is used in the ILC update law.

$$e^*(t) = \lim_{k \to \infty} (y_k - y_d) = (I - T_s(I - T_u + T_e T_s)^{-1} T_e) y_d(t) \qquad (1.12)$$

*defined on the interval* $(t_0, t_f)$, *if*

$$\|T_u - T_e T_s\|_i < 1. \qquad (1.13)$$

∎

Note that if $T_u = I$ then $\|e^*(t)\| = 0$ for all $t \in [t_o, t_f]$; otherwise the final converged error will be nonzero.

To understand the nature of ILC, consider the following:

**Question:** Given $T_s$, for the general linear ILC algorithm:

$$u_{k+1}(t) = T_u u_k(t) + T_e(y_d(t) - y_k(t)),$$

how do we pick $T_u$ and $T_e$ to make the final error $e^*(t)$ as "small" as possible?

**Answer:** Let the solution of the following problem be $T_n^*$:

$$\min_{T_n} \|(I - T_s T_n) y_d\|.$$

It turns out that we can specify $T_u$ and $T_e$ in terms of $T_n^*$ and the resulting learning controller converges to an optimal system input given by:

$$u^*(t) = T_n^* y_d(t).$$

This means that the essential effect of a properly designed learning controller is to produce the output of the best possible inverse of the system in the direction of $y_d$ [298]. This is the key characteristic of ILC.

Returning to the classic Arimoto-type ILC law (1.5), note that the basic formula for selecting the learning gain given in (1.6) does not require information about the system matrix $A$, which implies that ILC is an effective control scheme for improving the performance of uncertain (linear or nonlinear) dynamic systems. This is the main feature of ILC, as distinguished from classical control theories.

The ILC update rule of (1.5) is properly called a "D-type" ILC rule, as it operates on the derivative of the error. Likewise, we can consider PID-type ILC as given in (1.7), I-type ILC , or P-type ILC. For instance a P-type update rule (meaning no derivative and integral effects) can be written as

$$u_{k+1}(t) = u_k(t) + \gamma_k(t)(y_d(t) - y_k(t)), \qquad (1.14)$$

where $k$ is the iteration trial, $\gamma_k(t)$ is the proportional learning gain, $y_d(t)$ is the desired output, and $y_k(t)$ is the measured output. Note that this particular algorithm also has another feature: the gain is time-varying. This introduces another way to categorize ILC update laws: as time-invariant or time-varying.

As we noted above, we can also consider learning gains that are filters (e.g., the update law in (1.9)), which leads us to also consider the distinction between time-varying and time-invariant learning gain filters.

As we mentioned in a footnote above, ILC that only looks back at the most recent previous iteration is called first-order ILC. It is also possible to consider what is called higher-order ILC (HOILC), whereby data from more than one previous iteration is used. Consider, for instance, the ILC law

$$u_{k+1}(t) = u_k(t) + \sum_{i=k}^{i=k-l} \gamma_i(t)(y_d(t) - y_i(t)) \tag{1.15}$$

or the update algorithm

$$u_{k+1}(t) = \sum_{i=k}^{i=k-l} \lambda_i(t)u_i(t) + \sum_{i=k}^{i=k-l} \gamma_i(t)(y_d(t) - y_i(t)). \tag{1.16}$$

Similar to (1.15) and (1.16), which are both P-type rules, I-type, PD-type, and PID-type higher-order algorithms can be developed using the whole set of past control input signals and output error signals. [66] formulates perhaps the most general form of these type of algorithms, which is given as

$$u_{k+1} = \sum_{k=1}^{N}(I - \Lambda)P_k u_k + \Lambda u_o$$

$$+ \sum_{k=1}^{N}\left(\Phi_k e_{i-k+1} + \Gamma_k \dot{e}_{i-k+1} + \Psi_k \int e_{i-k+1}\mathrm{dt}\right). \tag{1.17}$$

It can be shown [66] that if $\sum_{k=1}^{N} P_k = I$, then by properly choosing the learning gain matrices we can ensure that $e_k$ converges to zero asymptotically.

So far, we have considered continuous time ILC algorithms. However, practically it is desirable to use the discrete-time system state-space model given below in (1.18–1.20), because microprocessor-based discrete, sampled-data systems are widely used in actual applications. Furthermore, since the nature of repetitive operations is finite-horizon, each iteration domain consists of finite number of discrete-time points, which can be represented by vectors (see (1.23–1.26) below). Thus, in the remainder of this monograph we will restrict our attention almost exclusively to discrete-time systems.

### 1.2.3 The Periodicity and Repetitiveness in ILC

In the descriptions of ILC given above, it has been implied that repetition in the system operation is with respect to time. However, more generally, the periodicity and the repetitiveness treated in ILC could be time-, state-, iteration-, or trajectory-dependent. Let us consider Figure 1.2(a) where the

mobile wheel is misaligned with the robot body. In this figure, the robot body $Y$ axis ($Y_b$) is not pointing in the same direction as the wheel $Y$ axis ($Y_w$). This misalignment ($\delta$) results in an eccentricity problem in the control system, which brings about an angle-dependent periodic disturbance. Thus, the eccentricity ($f_e$) is a function of angle $\theta$, i.e., $f_e = f(\theta)$. Figure 1.2(b) shows a satellite that rotates around the earth periodically. For a specified mission, the satellite is controlled to point in a particular direction. However, the satellite experiences external disturbances in the control system from the earth, sun, magnetic field, solar radiation, etc. These disturbances would be time-periodic, because the satellite orbit period is generally fixed. Figure 1.2(c) shows a robot manipulator which is time-periodic and whose initial condition at the start of each new period is same. The robot manipulator works time-periodically on a fixed trajectory (in this figure, by an angle $\theta$), but experiences iteration-varying disturbances. Thus, as shown in these figures, a periodic system could be defined as being in one of three different classes:

- Class A: state-periodic, but not time-periodic (Figure 1.2(a)).
- Class B: time-periodic, but not state-periodic (Figure 1.2(b)).
- Class C: state (starting state)-periodic and time-periodic (Figure 1.2(c)).

In more detail, in Figure 1.2(a), the eccentricity is the function of $\theta$ with the following relationship: $f_e(\theta) = f_e(\theta \pm 2n\pi)$ where $n$ is an integer; in Figure 1.2(b), the external disturbance $f_d$ is the function of $t$ with the following relationship: $f_d(t) = f_d(t \pm nT)$ where $T$ is the orbit period; in Figure 1.2(c), the external disturbance could be time-periodic but with the same initial states. Generally, iterative learning control treats the time-periodic system like Class C, whereas the periodic systems like Class A and Class B are studied under repetitive control (RC) and/or periodic control.[4] In this monograph, we will focus on Class C, although our framework can be easily modified to cover Class A and Class B.

### 1.2.4 Advantages of Using ILC

In the previous subsections, we discussed the characteristics of ILC, introduced basic ILC algorithms, and looked at different ways periodicity can occur. In this subsection, we present some advantages of ILC over typical control algorithms. These include:

- Precise trajectory tracking: If the four postulations given in Section 1.2.2 are satisfied, then a desired trajectory on a finite horizon in the time domain can be perfectly achieved. Thus, ILC algorithms can be effectively used for precise control in fields like semiconductor manufacturing

---

[4] However, Class A and Class B also have been widely studied in ILC. More or less, these days there is no distinction between the systems treated in ILC and the systems treated in RC. But, mathematical formulations of ILC and RC are different. This monograph handles these periodic systems under the ILC framework.
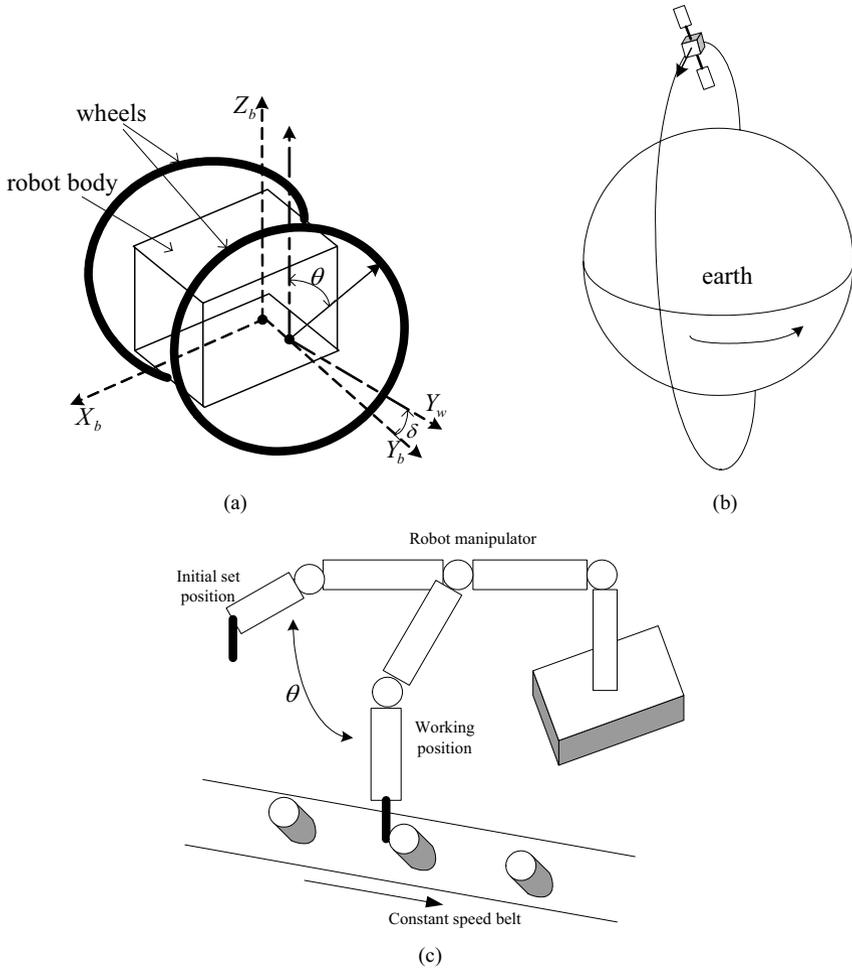
**Fig. 1.2.** (a): Eccentricity problem of a wheeled mobile robot. (b): Time-periodic disturbance in a satellite orbit. (c): Time-periodic robot manipulator in a manufacturing process, with an iteration-dependent disturbance.

    processes, robot arm manipulators, repetitive rotary systems, and factory batch processes.

- Monotonic convergence: The system convergence to a desired trajectory can be monotonic in iteration, if convergence conditions are met. Such monotonic convergence can prevent the break-up of hardware and high overshoots of the system trajectory.

- Controller design without accurate model information: An ILC controller can be designed without an accurate model of the system. The uncertainty that can be handled by ILC includes deterministic modeling error, para-

metric uncertainty, stochastic disturbances and noise, parameter variation, and deterministic external disturbances.

## 1.3 Research Motivation

The principal goal of this monograph is to investigate robustness issues in ILC from an iteration-domain perspective, with an objective of demonstrating analysis and design strategies that enable monotonic convergence and perfect trajectory tracking against a variety of uncertainties. Three main motivations for this study are provided in this section.

### 1.3.1 Motivation for Robust Interval Iterative Learning Control

Let us consider the following single-input, single-output (SISO), 2-dimensional system described in the state-space:

$$x_k(t+1) = Ax_k(t) + Bu_k(t) \qquad (1.18)$$
$$y_k(t) = Cx_k(t) \qquad (1.19)$$
$$x_k(0) = x_0, \qquad (1.20)$$

where $A \in \mathcal{R}^{n \times n}$, $B \in \mathcal{R}^{n \times 1}$, and $C \in \mathcal{R}^{1 \times n}$ are the matrices describing the system; $x_k(t) \in \mathcal{R}^n$, $u_k(t) \in \mathcal{R}$, and $y_k(t) \in \mathcal{R}$ are the state, input, and output vectors, respectively; $t$ represents discrete-time points along the time axis; and the subscript $k$ represents the iteration trial number along the iteration axis. Notice that $x_k(0) = x_0$ for all $k$. This is a key assumption in the ILC process and is called the initial reset condition (see again Section 1.2). Throughout this monograph, it is assumed that this initial reset condition is satisfied if there is no special indication otherwise.

For the system (1.18)–(1.20), from basic ILC theory (Theorem 1.1), a standard result is that the system is asymptotically stable (AS) if and only if $|1 - \gamma h_1| < 1$ with the ILC update law

$$u_{k+1}(t) = u_k(t) + \gamma e_k(t+1), \qquad (1.21)$$

where $h_1 = CB \neq 0$ (i.e., the system has relative degree 1) and $\gamma$ is the learning gain [302]. Thus, it is easy to design $\gamma$ such that the condition $|1 - \gamma h_1| < 1$ is satisfied, provided that $C$ and $B$ are known exactly. However, generally it is reasonable to assume that there exist model uncertainties in $C$ and $B$. In this case, it is necessary to select $\gamma$ considering all possible model uncertainties.

In fact, this kind of ILC problem has been studied widely in the literature. For example, refer to [522, 390, 293]. However, the existing research results are almost all restricted to the asymptotic stability (AS) problem. In ILC, it has been observed that AS may not be acceptable in a practical setting, because it is possible that an ILC system can experience very high overshoots of

the mean-square error during the transient before the system converges [279]. Thus, the monotonically-convergent ILC (MC-ILC) design problem has been considered to be a practically important issue, as claimed and demonstrated in [415, 279].[5] With regard to monotonically-convergent ILC, numerous publications are available. An approximate monotonic decay condition was given in [417]. A monotonic convergence condition through parameter optimization was introduced in [340]. Monotonic convergence of a Hamiltonian system was guaranteed in [140]. A maximum singular value-based monotonic convergence condition was given in [308, 331]. Following these works, in this monograph, we are thus motivated to consider not only robust asymptotic stability against interval perturbations in the system Markov parameters, but also robust monotonic convergence (i.e., exponential stability in the iteration domain).

To design a monotonically-convergent ILC algorithm, many results in the literature require information about the $A$ matrix [298, 363, 346, 136, 9]. Though most results have considered only the nominal plant, it is again natural to consider $A$ to be an uncertain matrix. Thus we are motivated to consider the design of the learning gain matrix when considering model uncertainties in $A$, $B$, and $C$.

There are a number of analysis methodologies that have been used in the ILC literature. In one such method the 2-dimensional problem of (1.18)–(1.20) is reformulated into a 1-dimensional problem. This is called the super-vector ILC (SVILC) framework [298] and it is in this framework that we will consider the robust MC-ILC problem.

To describe the SVILC methodology, take $z$-transforms (in time) of (1.18)–(1.20) and define the resulting plant to be $G(z)$.[6] This gives

$$Y(z) = G(z)U(z)$$
$$= (h_m z^{-m} + h_{m+1} z^{-(m+1)} + h_{m+2} z^{-(m+2)} + \cdots)U(z), \quad (1.22)$$

where $m$ is the relative degree of the system, $z^{-1}$ is the delay operator in the discrete-time domain, and the parameters $h_i$ are Markov parameters of the impulse response system of the plant $G(z)$. If we now define the following vectors:[7]

$$U_k = (u_k(0), u_k(1), \ldots, u_k(N-1)) \qquad (1.23)$$
$$Y_k = (y_k(m), y_k(m+1), \ldots, y_k(N-1+m)) \qquad (1.24)$$
$$Y_d = (y_d(m), y_d(m+1), \ldots, y_d(N-1+m)) \qquad (1.25)$$
$$E_k = Y_d - Y_k = (E_k(m), E_k(m+1), \ldots, E_k(N-1+m)), \qquad (1.26)$$

then the linear plant can be described by $Y_k = HU_k$, where $H$ is a Toeplitz matrix of rank $N$ whose elements are the Markov parameters of the plant

---

[5] For more precise definitions of AS and MC, see Section 3.2 and Section 4.1.

[6] $G(z) = C(zI - A)^{-1}B + D$.

[7] The process of forming "super-vectors" $U_k$, $Y_k$, $Y_d$, and $E_k$ is called "lifting" in the literature [298].

$G(z)$, given by:

$$H = \begin{bmatrix} h_m & 0 & 0 & \dots & 0 \\ h_{m+1} & h_m & 0 & \dots & 0 \\ h_{m+2} & h_{m+1} & h_m & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{m+N-1} & h_{m+N-2} & h_{m+N-3} & \dots & h_m \end{bmatrix}. \qquad (1.27)$$

Throughout this monograph, this matrix is called the system Markov matrix or simply the Markov matrix.

The monotonic convergence condition for the system (1.18)–(1.20) with the standard Arimoto-type ILC update law (1.21) is now simply given as

$$\|I - H\Gamma\| < 1,$$

where $\| \cdot \|$ is an operator norm and $\Gamma$ is the learning gain matrix. $\Gamma$ is a diagonal matrix whose diagonal elements are the scalar learning gains $\gamma$.

In the super-vector framework we just described, the system given in (1.18)–(1.20) is assumed to be the nominal plant, i.e., neither model uncertainty nor process disturbances or measurement noises are considered. Thus, the MC condition $\|I - H\Gamma\| < 1$ is not practically meaningful when taking into account model uncertainties or external disturbances and noise. To address this, consider instead the following 2-dimensional uncertain plant model:

$$x_k(t+1) = (A + \Delta A)x_k(t) + (B + \Delta B)u_k(t) + v(k,t) \qquad (1.28)$$
$$y_k(t) = (C + \Delta C)x_k(t) + w(k,t), \qquad (1.29)$$

where $\Delta A$, $\Delta B$, and $\Delta C$ are model uncertainties, and $v(k,t)$ and $w(k,t)$ are time- and iteration-dependent process disturbance and measurement noise signals, respectively. Recall that $t$ is the discrete-time point along the time axis, which means that $t$ is defined in a finite interval. That is, on each iteration or trial, $t$ has $N$ different discrete-time points. Meanwhile, $k$ is defined on an infinite horizon, so it increases monotonically. Thus we can consider two different types of model uncertainties. The first type is iteration-independent model uncertainty, while the second type is iteration-dependent model uncertainty.

We are interested in addressing the robustness and convergence properties of systems such as (1.28)–(1.29). However, the SVILC framework requires analysis based on the system Markov matrix. Thus, it becomes necessary to convert the model uncertainty of (1.28)–(1.29) to uncertainty associated with the Markov matrix (1.27). To our knowledge, this uncertainty conversion problem has never been addressed in the existing literature except in our own work. This problem, which we call "interval model conversion," will be carefully addressed in this monograph. In particular, for the interval model conversion problem, the power of an interval matrix will need to be computed and we will show how this can be done in a computationally efficient and non-conservative fashion.

Next, let us suppose that Markov matrix includes the model uncertainty converted from the uncertain plant (1.28)–(1.29) and let us denote the uncertain system Markov matrix as $H = H^o + \Delta H$, where $H^o$ is the system Markov matrix corresponding to the nominal plant and $\Delta H$ is the uncertain system Markov matrix corresponding to the uncertainty of the uncertain plant. Then, our task is to find a learning gain matrix such that the system is AS or MC against all possible uncertain plants $H$. There are two issues with regard to this robust ILC design problem. The first issue is related to the conservativeness and the computational cost of the proposed method, and the second issue is related to the performance of the algorithm. The performance issue is concerned with the stability types: asymptotic stability (AS) and monotonic convergence (MC).[8]  As noted, although there are numerous results on asymptotic and robust stability in ILC [522, 390, 183], to date there is no systematic analysis and synthesis framework addressing robust monotonically-convergent ILC. Most existing works have focused on asymptotic stability for plants with model uncertainty described in the state-space form given by (1.28)–(1.29). To our knowledge, outside of our own work, no systematic approach for handling the uncertainty in $H$ (i.e., $\Delta H$) has been reported. Furthermore, though in many existing works, optimal-ILC [9, 181, 342, 341], stochastic noise [397, 396, 395, 54], and frequency-dependent uncertainty [360] have been considered, these techniques can give conservative results. We show, however, that using parametric interval concepts can reduce the conservatism connected with robustness tests and can provide tighter monotonic convergence conditions.

In summary, there are three main research motivations for studying the robust interval ILC problem. The first motivation is to solve fundamental problems associated with the robustness of ILC designs when the plant is subject to parametric uncertainty. Then, based on these results, the second task is to guarantee monotonic convergence against all possible interval uncertainties. Finally, the goal is to reduce the conservatism related to robust stability tests.

## 1.3.2 Motivation for $H_\infty$ Iterative Learning Control

From the ILC literature, there is no systematic approach to handling iteration-varying model uncertainty, iteration-varying external disturbances, or iteration-varying stochastic noise all together [360]. Even though time-domain-based

---

[8] Note that stability in an ILC problem refers to the boundedness of signals at fixed points of time, considered along the iteration axis. By assumption (due to the finite-time horizon), traditional stability along the time axis is achieved by default (except, perhaps, in the case of a nonlinear system exhibiting finite escape-time behaviors). Thus, by AS, we mean that the ILC system converges to the desired trajectory as the iteration number increases. By MC, we mean that the ILC system converges without overshoot to the desired trajectory as iteration number increases.

$H_\infty$ ILC schemes and 2-dimensional approaches have been suggested for making a unified ILC framework, research to date has been limited to iteration-independent uncertainty and disturbances. However, if we can cast the supervector notation as defined by (1.23)–(1.26) into a traditional discrete-time $H_\infty$ framework (where now "time" is actually "iteration"), we can obtain a unified robust control framework on the iteration domain. Furthermore, our proposed unified robust $H_\infty$ ILC approach on the iteration domain provides a way to consider and discuss frequency-domain analysis on the iteration axis. Recall that in ILC the time axis is finite-horizon. Thus, the corresponding frequency domain transformed from the time axis should also be finite. However, in control engineering the frequency domain is usually infinite. Thus, the frequency-domain-based ILC analysis, transformed from the finite time axis, is not suitable from an analytical perspective. However, in our proposed $H_\infty$ scheme on the iteration axis, the discrete infinite iteration axis is readily transformed into the discrete infinite frequency domain. Hence, the $H_\infty$ ILC scheme on the iteration axis provides a unified robust control framework on the infinite frequency domain that is analytically correct.

### 1.3.3 Motivation for Stochastic Iterative Learning Control

Even though the $H_\infty$ ILC scheme on the iteration domain presented in this monograph provides a unified framework, it is not related to monotonic convergence. Motivated by this observation, we can raise a question: is it possible to design a learning gain matrix to ensure monotonic convergence when considering stochastic noise or iteration-varying model uncertainty? Further, a related question is how to analytically estimate the steady-state error that the ILC system can achieve under stochastic noise and model uncertainty. In the stochastic ILC approach proposed in this monograph, we try to estimate the ultimate baseline error of an uncertain ILC system over which monotonic convergence is guaranteed. And, we wish to make this estimate in an off-line manner. A related problem also arises in the area of networked-control systems (NCS), where data dropout problems have been popularly studied. In this monograph, we try to integrate the NCS into an ILC framework so that an overall intermittent ILC system can be developed that is robust against extreme data dropout situations in a network.

## 1.4 Original Contributions of the Monograph

This monograph makes the following theoretical contributions:

- Conditions for robust stability in the iteration domain are provided for parametric interval systems.
- Techniques for converting from time-domain interval models to Markov interval models are given.

- A monotonically-convergent ILC system is designed under parametric interval uncertainties and/or stochastic noise considerations.
- Robust $H_\infty$ ILC is designed on the iteration domain, taking into account three different types of uncertainties: iteration-variant/invariant model uncertainty, external disturbances, and stochastic noise.
- The baseline error of the ILC process is analytically established, which provides a novel idea for designing the ILC learning gain matrix in an off-line manner.
- Solutions for three fundamental interval computational problems are offered: robust stability of an interval polynomial matrix system, the power of an interval matrix, and the maximum singular value of an interval matrix.

It is the main contribution of this monograph to provide new analytical tools for designing robust ILC systems. Using the super-vector approach, the robustness problem of ILC is discussed purely on the iteration domain. Parametric interval uncertainty enables us to design both the monotonically-convergent ILC process and a less conservative robust ILC system. Indeed, this robust, monotonically-convergent ILC design method makes a significant contribution to practical ILC applications because it avoids unacceptable overshoot on the iteration domain while considering all possible models the controller might face. Furthermore, by casting the $H_\infty$ framework and Kalman filtering into the SVILC framework, the monograph provides a different design perspective for stochastic and frequency-domain uncertain ILC systems than has typically been found in the literature. Additionally, analytical solutions for the three fundamental interval computational problems mentioned above are provided. These solutions can be effectively used for solving various types of control and systems problems. For example, robust controllability, robust observability, multi-input multi-output robust control theory, robust monotonically-convergent stability problems, and robust boundary calculations for the model conversion problem can all be addressed using the results presented in the monograph.

Iterative Learning Control
Robustness and Monotonic Convergence for Interval
Systems
Ahn, H.-S.; Moore, K.L.; Chen, Y.