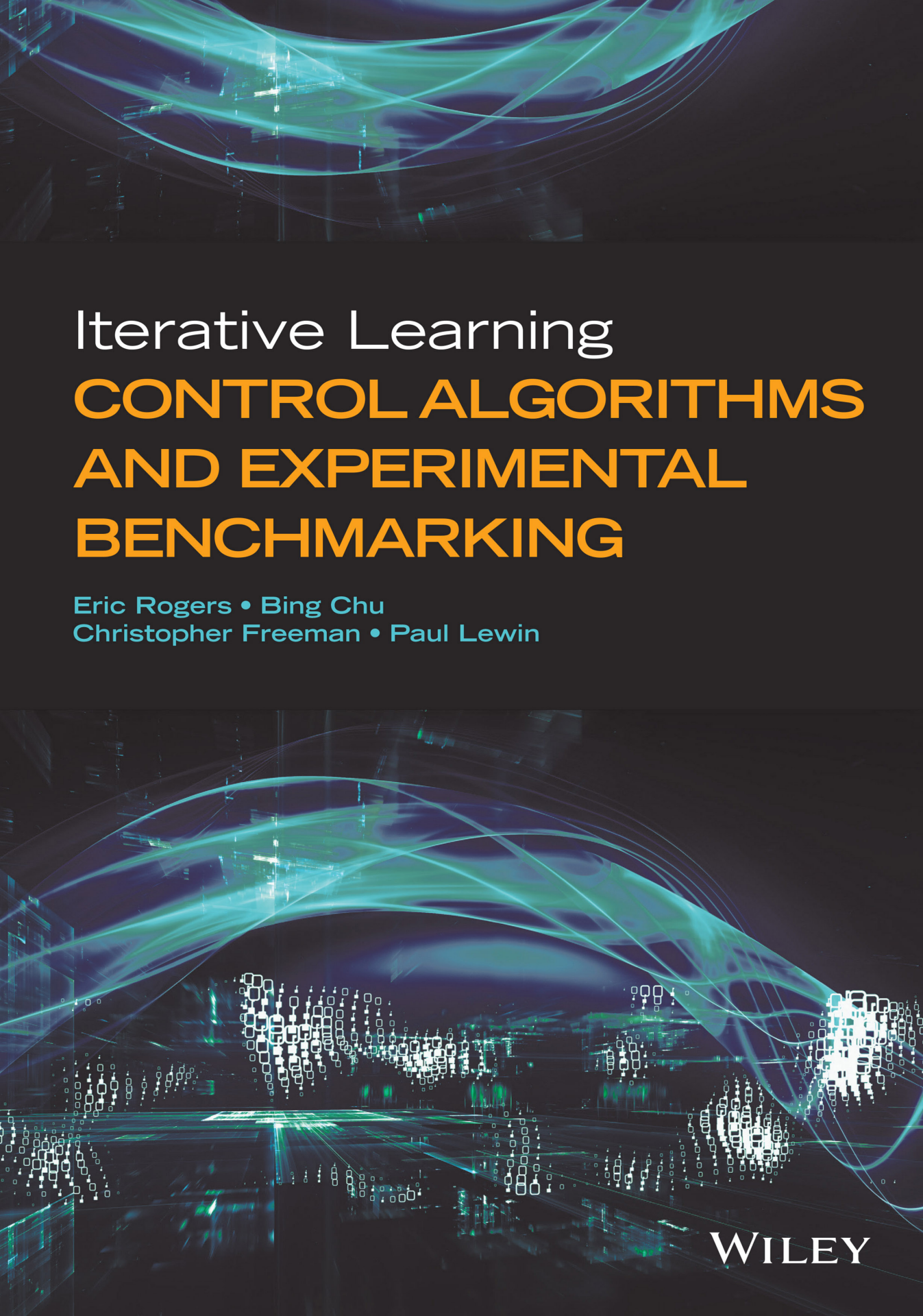


Iterative Learning CONTROL ALGORITHMS AND EXPERIMENTAL BENCHMARKING

Eric Rogers • Bing Chu
Christopher Freeman • Paul Lewin

WILEY

Iterative Learning Control Algorithms and Experimental Benchmarking

Eric Rogers, Bing Chu, Christopher Freeman and Paul Lewin
University of Southampton, UK

WILEY

This edition first published 2023

Registered Office
John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, USA
John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, UK

For details of our global editorial offices, customer services, and more information about Wiley products visit us at www.wiley.com.

Wiley also publishes its books in a variety of electronic formats and by print-on-demand. Some content that appears in standard print versions of this book may not be available in other formats.

Library of Congress Cataloging-in-Publication Data Applied for:

ISBN 9780470745045 (hardback)

Cover Design: Wiley
Cover Image: © agsandrew/Getty Images

Set in 9.5/12.5pt STIXTwoText by Straive, Chennai, India

Contents

Preface

Humans and other species seeking to complete many finite duration tasks proceed by an attempt at it. If not successful, then aim to learn from the outcome to complete the task after many repeated attempts if necessary. This approach led Japanese robotics researcher Suguru Arimoto to say, "It is human to make mistakes, but it also human to learn from such experience. Is it possible to think of a way to implement such a learning ability in the automatic operation of dynamic systems?" This statement is widely seen as the starting point for iterative learning control (ILC), which is applicable, in its basic form, to systems that complete the same finite duration task over and over again. The sequence of operations is that the task is executed, and then the system resets to the starting position, ready for the next in the sequence. As in many areas, the terminology in ILC varies between terming an execution a trial or an iteration, or a pass. In this book, a trial is the default terminology, and the associated finite duration of the operation is termed the trial length.

The standard ILC design specification assumes that a reference trajectory is available. The difference between the output on any trial and the reference trajectory is the error on this trial. Then the task is to design a control law that forces the sequence of errors to converge from trial-to-trial, ideally to zero or to within an acceptable tolerance. The convergence of ILC designs is a very well-studied problem for various forms of dynamics. It has led to many design algorithms and, in many cases, at least laboratory-level supporting experimental evaluation.

Robotics is a source of many problems to which ILC can be applied. A particular example is "pick and place" operations, e.g. a robot required to collect an object from a given location, transfer it over a finite time, place it on a conveyor under synchronization, and then return to the starting location for the next one, and so on. The control input for the subsequent trial can be computed during the reset time, with all of the previous trial error data available if required since the data are generated on the previous trial. In this application, error convergence in the trial variable must be supported by control of the dynamics generated along the trials, e.g. to prevent unacceptable behavior, such as oscillations that would be of particular concern should the object be an open-necked container containing a liquid.

This book focuses on ILC analysis and design with experimental benchmarking. The analysis and design methods coverage include linear and nonlinear dynamics, time-varying, and stochastic dynamics. Both linear and nonlinear designs are considered, where one design for the latter case is for examples where actuation saturation is a possibility.

Two-dimensional (2D) systems, i.e. information propagation in two independent directions, are a well-established area of research where image- and signal-processing applications are not the case for control. Repetitive processes are a distinct class of 2D systems where information propagation occurs only over a finite duration in one of the independent directions. Moreover, this is a feature of the dynamics and not an assumption introduced for analysis purposes. These processes are a

natural fit for ILC dynamics. This book develops a range of repetitive process-based designs. These designs complement/improve those based, e.g. on the lifting approach where the finite trial length allows the trial-to-trial error dynamics to be represented by a standard linear systems difference equation. Hence, results for this latter case can be applied.

A particular feature of this book is algorithm, or control law, development supported by at least laboratory experimental benchmarking. The latter aspect contains results from experimental testbeds, such as a laboratory-based system replicating the pick-and-place task for robotics, and also from implementation on physical systems such as a free-electron laser system. In some cases, a laboratory testbed has been specially designed for ILC experiments, e.g. a servomechanism and a rack feeder system.

Another distinctive feature is the application of ILC in healthcare. This area is represented by robotic-assisted upper-limb stroke rehabilitation. In physiotherapy, the recommended approach to rehabilitation after a stroke is based on repeated practice of a task where experience from previous attempts is used to update the effort applied on the next attempt. Still, the difficulty is that the patient cannot move the affected limb and therefore does not get feedback. In such cases, functional electrical stimulation, appropriately designed, can be applied to assist movement.

In a program of research stretching back to the mid-2000s, collaborative research at the University of Southampton, between physiotherapists and control engineers, has used ILC to regulate the level of assistive stimulation applied. This research has followed through with small-scale clinical trials with patients. The critical feature is that if the patient improves with each successive attempt, the level of applied simulation goes down. Therefore, the patient's voluntary effort is increasing. This feature is essential in rehabilitation, where the ultimate aim is to restore independent living. A particular feature of this book is the use of the rehabilitation application to highlight the application of ILC designs, ranging from the tuning of simple structure controllers to nonlinear model-based designs.

The first two chapters give background to the subject area, including the basics of the settings used for analysis and design and the models for some physical examples used to validate the designs, respectively, experimentally. Chapter 3 gives an overview of design for performance. Chapter 4 covers the design of control laws with simple structures and supporting experimental results. Chapters 5–7 cover design based on linear models using various settings, including the repetitive processes/2D systems setting and robust design, with supporting experimental results. Chapter 8 begins the coverage of nonlinear effects, focusing on input constraints. Two designs are developed, one of which uses the repetitive process setting for the case of a saturating input and the other constrained design for linear time-varying dynamics, again with experimental results. Chapter 9 then addresses ILC for distributed parameter systems, with particular emphasis on building approximate dynamics models, and one of the designs is supported by experimental results.

This book has taken a long time to complete and has benefited hugely from collaboration with others. The first and second authors have been working with David Owens (University of Sheffield, UK) since their PhD studies, and results from this collaboration have contributed significantly to several chapters. A very long-standing collaboration with Krzysztof Galkowski, Wojciech Paszke, and Lukasz Hladowski (University of Zielona Gora, Poland) and Slawek Mandra (Nicolaus Copernicus University in Toruń, Poland) forms the basis for the results on repetitive process-based design and experimental validation for linear systems. The extension to design based on nonlinear repetitive process stability theory has results from a collaboration with Pavel Pakshin and Julia Emelianova (R.E. Alekseev Nizhny Novgorod State Technical University, Russia). The experimental results for constrained design for linear time-varying systems in Chapter 8 and the

those in Chapter 9 for distributed parameter systems were obtained from experimental testbeds designed and commissioned by Harald Aschemann and Andreas Rauh (University of Rostock, Germany). The data-driven model example in Chapter 13 arose from collaboration with Simon Vestergaard Johansen and Jan Dimon Bendtsen (Aalborg University, Denmark) and the quantum control application from collaboration with Re-Bing Wu (Tsinghua University, PR China).

In the University of Southampton, our completed PhD students Dr. James Ratcliffe, Dr. Jack Cai, and Dr. Weronika Nowicka contributed to the design and commissioning of the gantry robot test facility, the reference shift algorithm of Chapter 4, and the wind turbine application of Chapter 9, respectively. The contributions of Professor Owen Tutty to the flow modeling for the wind turbine application are also acknowledged. A chance conversation at a university event with Professor Jane Burridge led to the start of a long and ongoing research program on using ILC in robotic-assisted stroke rehabilitation with her, Dr. Ann-Marie Hughes, and Dr. Katie Meadmore.

University of Southampton, UK
December 2002 *Eric Rogers, Bing Chu, Christopher Freeman, and Paul Lewin*

1

Iterative Learning Control: Origins and General Overview

A commonly encountered requirement in some industries is for a machine to repeat the same finite duration operation over and over again. The exact sequence is that the procedure is completed and then the system or process involved resets to the starting location and the next one begins. A typical scenario is a gantry robot, such as the one shown in Figure 1.1, undertaking a "pick and place" operation encountered in many industries where the following steps must be conducted in synchronization with a conveyor system: (i) collect an object from a fixed location, (ii) transfer it over a finite duration, (iii) place it on the moving conveyor, (iv) return to the original location for the next object and then, (v) repeat the previous four steps for as many objects as required or can be transferred before it is necessary to stop for maintenance or other reasons. Stopping these robots for such reasons in high-throughput applications means down time and lost production.

Figure 1.2 shows a 3D reference trajectory for the gantry robot and Figure 1.3 its X-axis component. On each execution a variable, say y, is defined over the finite duration taken to move from the pick to place locations, e.g. $y(t),\ 0 \leq t \leq \alpha < \infty$, but it is also required to distinguish variables according to which execution is under consideration. One option, used except where stated otherwise in this book, is to write $y_k(t),\ 0 \leq t \leq \alpha$, where k is a nonnegative integer termed the trial number with α denoting the trial duration or length.

Let $r(t)$ be a prespecified 3D path or trajectory that the robot is required to follow between the "pick" and "place" locations (and back to the "pick" location), such as Figure 1.2 or, to focus on one axis, Figure 1.3. Then on trial k, the error is $e_k(t) = r(t) - y_k(t)$ and if $e_k(t) \neq 0$, the question is: how should the control input signal be adjusted to reduce or remove this error?

In applications such as the one considered, the system is performing the same operation repeatedly under the same operating conditions. One approach to control design is to copy human behavior and aim to learn from experience, i.e. the errors generated on previous trials are rich in information. This information is not exploited by a standard controller that would produce the same error on each trial. Iterative learning control (ILC) aims to improve performance by using information from previous trials to update the control law to be applied on the next one. As the above example illustrates, a significant application area for ILC is robotics.

As with other areas within control systems, there has been a debate, see the next section, on the origins of ILC. Since the 1980s, when concentrated research began, applications for ILC have spread beyond robotics in the industrial domain and outside engineering into healthcare. An example from the latter area, in the form of robotic-assisted upper-limb stroke rehabilitation, is introduced in Section 2.4 and considered in depth in Sections 4.2, 10.2, 10.3, and 11.4 of this book.

Iterative Learning Control Algorithms and Experimental Benchmarking, First Edition.
Eric Rogers, Bing Chu, Christopher Freeman and Paul Lewin.

Figure 1.1 A gantry robot for a pick-and-place operation with the axes marked.

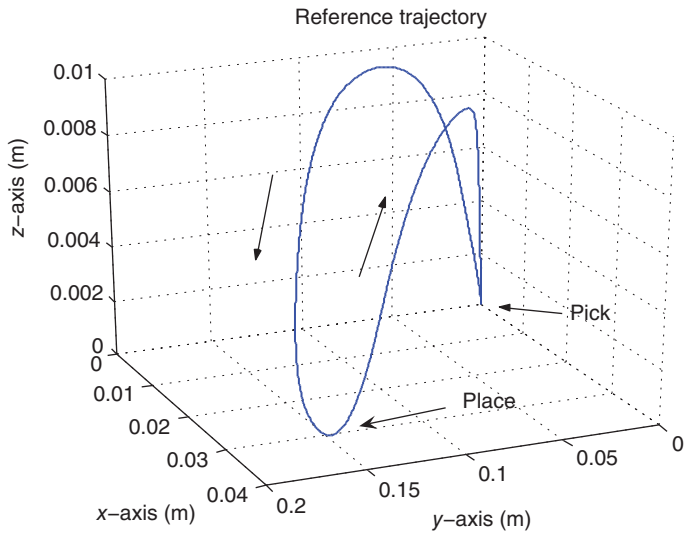


Figure 1.2 3D reference trajectory for the gantry robot.

1.1 The Origins of ILC

According to [2, 33] and others, the basic idea of ILC was first proposed in a 1971 patent [89] and the journal article [256] published in Japanese. The first concerted volume of work that initiated widespread interest was, in particular, the journal paper [12], which considers a simple first-order linear servomechanism system for speed control of a voltage-controlled DC-servomotor. In this section, this system is used to highlight the essence of ILC, and it is appropriate to start by quoting parts of the opening two paragraphs in this paper.

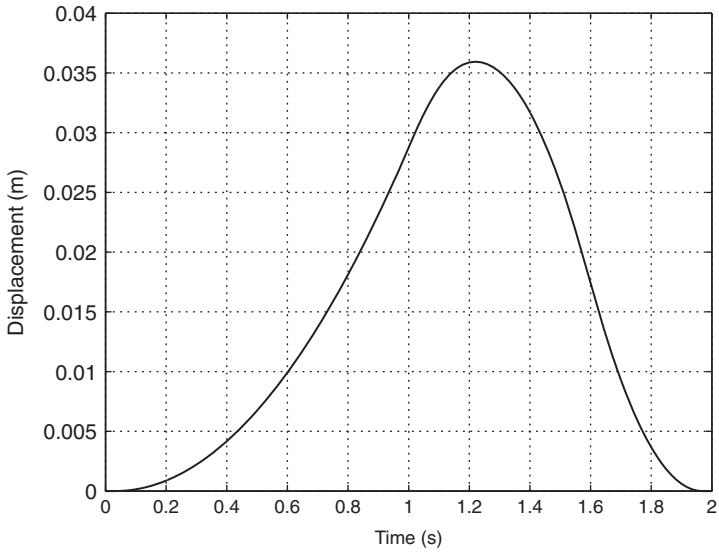


Figure 1.3 *X*-axis component of the gantry robot reference trajectory.

"It is human to make mistakes, but it also human to learn from such experience. Athletes have improved their form of body motion by learning through repeated training and skilled hands have mastered the operation of machines or plants by acquiring skill in practice and gaining knowledge. Is it possible to think of a way to implement such a learning ability in the automatic operation of dynamic systems?"

Motivated by this consideration, Arimoto et al. [12] proposed "a practical approach to the problem of bettering the present operation of mechanical robots by using the data of previous operations." This work constructed an "iterative betterment process for the dynamics of robots so that the trajectory of their motion approaches asymptotically a given desired trajectory as the number of operation trials increases." The example in [12] is next used to illustrate the construction of ILC laws and the behavior that can arise. A critical feature is "the direct use of the underlying dynamics of the objective systems."

The form of the control law developed in [12] applies to systems that are required to track a desired reference trajectory **of a fixed length** α and **specified a priori**. After each trial, **resetting of the system states** occurs, during which time the **measured output** is used in the construction of the control input for application on the next trial. In [12], the system dynamics were assumed **trial-invariant** and **invertible**. These six distinguishing features of ILC highlighted in bold provided the basis for a major area of research in the control systems community internationally, both in terms of theory and an ever-broadening list of applications, many with supporting experimental verification or actual implementation.

As a motivating example, Arimoto et al. [12] considered speed control of a voltage-controlled DC servomotor where if the armature inductance is sufficiently small and mechanical friction is ignored, the resulting controlled dynamics are described by

$$\dot{y}(t) + ay(t) = bv(t) \tag{1.1}$$

where $\dot{y}(t)$ denotes the angular velocity of the motor, $v(t)$ is the input voltage, and a and b are constants.

Suppose that a reference signal, or trajectory, $r(t)$, for the angular velocity $y(t)$ is given over the fixed finite duration $0 \le t \le \alpha$ and also that the system dynamics are unknown in the sense that the exact values of a and b in (1.1) may not be available. Also, it is assumed that $r(t)$ is continuously differentiable, which is a commonly used assumption in differential ILC design.

The design problem is to construct an input voltage $v(t)$ that coincides with $r(t)$ over $0 \le t \le \alpha$. If an arbitrary input $v_0(t)$ is applied, the error between the desired output $r(t)$ and the first response $y_0(t)$ is

$$e_0(t) = r(t) - y_0(t) \tag{1.2}$$

where

$$\dot{y}_0(t) + a y_0(t) = b v_0(t) \tag{1.3}$$

Also, construct the voltage

$$v_1(t) = a v_0(t) + b^{-1} \dot{e}_0(t) \tag{1.4}$$

and apply this as the input on the next trial. Storing the resulting error $e_1(t)$, constructing $e_2(t)$, and continuing this sequence of operations give on trial $k+1$

$$\begin{aligned} v_{k+1}(t) &= v_k(t) + b^{-1} \dot{e}_k(t) \\ e_k(t) &= r(t) - y_k(t) \\ y_k(t) &= e^{-at} y_k(0) + b \int_0^t e^{-a(t-\tau)} v_k(\tau) d\tau \end{aligned} \tag{1.5}$$

Suppose that $y_0(0) = r(0) - y_k(0)$ for all k. Then since $r(t)$ is continuously differentiable and using (1.5) gives

$$\dot{e}_k(t) = a \int_0^t e^{-a(t-\tau)} \dot{e}_{k-1}(\tau) d\tau \tag{1.6}$$

Hence,

$$\begin{aligned} |\dot{e}_k(t)| &\le |a|^2 \int_0^t dt_1 \int_0^{t_1} e^{-a(t-t_2)} |\dot{e}_{k-2}(t_2)| dt_2 \\ &\le |a|^k \int_0^t \int_0^{t_1} \cdots \int_0^{t_{k-1}} |\dot{e}_0(t_k)| dt_k dt_{k-1} \cdots dt_1 \\ &\le \frac{\mathscr{K} |a|^k \alpha^k}{k!} \to 0, \quad k \to \infty \end{aligned} \tag{1.7}$$

where

$$\mathscr{K} = \max_{0 \le t \le \alpha} |\dot{e}_0(t)|$$

Supporting simulation studies are in [12]. Since this first work, various definitions of ILC have been given in the literature, including the following quoted in [2]:

1. 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 [12].
2. It is a recursive online control method that relies on less calculation and less a priori knowledge about the system dynamics. The idea is to apply a simple algorithm repetitively to an unknown system, until perfect tracking is achieved [26].
3. ILC is an approach to improve the transient response of the system that operates repetitively over a fixed time interval [166].

4. ILC considers systems that repetitively perform the same task with a view to sequentially improving accuracy [4].
5. ILC is to utilize the system repetitions as an experience to improve the system control performance even under incomplete knowledge of the system to be controlled [50].
6. The controller 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 [210].
7. 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.
8. 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 core activity in ILC research, which is the construction and subsequent analysis of algorithms [266].

Each of these definitions has their focus, but the underlying question is the same: how to improve performance using information from previous trials to update the control law applied on the current one? In some applications, ILC will form only one possible way of designing the controller to be employed. Therefore, it is essential to understand how ILC differs from other forms of control. One place to start is with other forms of control that can be categorized as using some form of learning.

Learning-type control algorithms include adaptive control, neural networks, and repetitive control (RC), where an adaptive control solution for a problem aims to modify the controller, whereas ILC adjusts the control input. Hence, these approaches differ since the controller is a system, and the control input is a signal. In most cases, adaptive controllers do not take advantage of the information contained in repetitive reference or command signals.

In a healthcare application of ILC [84], see Section 2.4 for a general introduction with a more detailed treatment in Chapters 4, 10, and 11, to robotic-assisted stroke rehabilitation, the control signal is electrical stimulation. If a patient is improving, then the level of electrical stimulation applied must decrease, i.e. control of the input signal is of paramount importance. Neural network learning in a control system problem involves modifying the parameters in the controller and not the control signal.

In many respects, ILC is closest to RC, where in the latter [103] there is continuous operation, and the initial conditions at the start of any trial are those at the end of the previous trial, and there is no resetting. An RC counterpart problem to the robotic pick-and-place operation of the last section, for which ILC is suitable, is the control of the read/write head of a hard disk drive, where each trial is a full rotation of the disk, and the next one immediately follows the completion of the current one. The critical difference between ILC and RC is the resetting of the initial conditions for each trial. Still, it can be shown that a duality exists between them under certain conditions, see, e.g. [78].

1.2 A Synopsis of the Literature

Research in ILC can, as in almost all other areas of control systems research, be broadly broken down into developments in the underlying systems theory and also applications with some work combining both in the form of theory followed by experimental validation. In the theoretical/algorithm development research, several of the six postulates given in Section 1.1 have been relaxed in recent years, but the concept of learning from experience gained over repetitions of a

task remains unchanged. The mature algorithmic development, especially for linear systems, has, in turn, led to increased effort on extending the boundaries of achievable performance, together with new analysis and design methods often inspired by practical applications.

In the case of nonlinear systems, the primary focus of applying ILC designed using a linear model approximation of the dynamics has shifted toward practical control problems and associated theoretical challenges. Also, interest in the development of nonlinear model-based ILC has significantly increased, again driven by practical applications. These trends can be seen in the survey papers [2, 33] and papers published in learned journals, conferences, and research monographs since the publication of these two surveys.

Since the publication of [12], ILC has broadened in breadth and depth, fusing with established fields, such as robust, adaptive, and optimal control, together with neural networks, fuzzy logic, and many others. Application areas have also expanded, most heavily in robotics, rotary systems, process control, bioapplications, power systems, and semiconductors and, more recently, a transfer to other areas including healthcare. While ILC is still a relatively young field, with a combined body of research work amassing around 1/30th of more established areas, there is a strong linear trend of increasing year-on-year publications on both theory and applications.

The relative merits of ILC and feedback and feedforward design have been considered in the literature. For example, in [33] it has been asserted that ILC has advantages for some problem classes but cannot be applied to all control problems, and the question is how does ILC compare with well-designed feedback and feedforward control? This question is wide-ranging, but the following general remarks [33] can be made.

First, the goal of ILC in the basic case is to generate a feedforward control that tracks a specified reference or rejects a repeating disturbance. In contrast, a feedback controller reacts to inputs and disturbances, and hence, there is always a lag in transient tracking. Feedforward control can eliminate this lag for known or measurable signals but typically not for disturbances. Also, ILC is anticipatory and can compensate for exogenous signals, such as repeating disturbances, in advance by learning and using information from previous trials and does not require that the exogenous signals, either reference or disturbances, be known or measured. Such signals are, however, required to repeat from trial to trial. Moreover, a feedback controller can deal with variations or uncertainties in the system model, i.e. robust control, but the performance of a feedforward controller depends critically on how accurately the system is known, and unmodeled nonlinear behavior and disturbances can restrict the effectiveness of feedforward control.

Noise and nonrepeating disturbances degrade tracking performance, and in common with feedback control, sensitivity to noise can be mitigated by an observer. Again, however, performance depends on to what extent the system dynamics are known. The trial-to-trial structure of ILC and, in particular, the reset between trials allows for advanced filtering and signal processing. In particular, zero-phase noncausal filtering can be used for high-frequency attenuation without introducing phase lag. To reject nonrepeating disturbances, a feedback controller in combination with ILC is advantageous.

The relative merits of ILC against alternatives will be discussed at many points in this book, and in the next section, representations or models widely used in ILC research are introduced, together with some commonly used control law structures.

1.3 Linear Models and Control Structures

As noted in the introduction to this chapter, there is no uniformity of notation in the ILC literature. The notation for continuous-time variables is in the previous section. For discrete variables, the

notation is $z_k(p),\ 0 \le p \le N-1,\ k \ge 0$, where z denotes the vector or scalar valued variable under consideration, N denotes the number of samples along a trial and, hence, multiplying this number by the sampling period gives the trial length. This notation is used throughout this book except where stated otherwise.

Analysis of ILC schemes can begin in either the continuous or discrete-time settings, but a large number of actual implementations will be in the latter. This section considers the continuous-time, or differential, dynamics next.

1.3.1 Differential Linear Dynamics

If the system considered is represented by a linear time-invariant model, the state-space model in the ILC setting is

$$\begin{aligned} \dot{x}_{k+1}(t) &= Ax_{k+1}(t) + Bu_{k+1}(t) \\ y_{k+1}(t) &= Cx_{k+1}(t), \quad 0 \le t \le \alpha, \quad k \ge 0 \end{aligned} \tag{1.8}$$

where on trial k, $x_k(t) \in \mathbb{R}^n$ is the state vector, $y_k(t) \in \mathbb{R}^m$ is the output vector, and $u_k(t) \in \mathbb{R}^l$ is the control input vector. The control task is to force $y_k(t)$ to track a supplied reference signal $r(t)$ over the fixed trial length $0 \le t \le \alpha$, and it is assumed that (i) trial length is the same for all trials, (ii) the state initial vector is of the form $x_{k+1}(0) = d_{k+1},\ k \ge 0$, where the entries in d_{k+1} are known constants, and (iii) the system dynamics are time-invariant and deterministic, i.e. noise-free. As discussed at the relevant places in this book, these assumptions can be relaxed. On trial k, $e_k(t) = r(t) - y_k(t),\ 0 \le t \le \alpha$, and the control law used in [12] for differential linear time-invariant systems, given in this chapter as the first equation in (1.5), is a particular case of

$$u_{k+1}(t) = u_k(t) + K_d\dot{e}_k(t) \tag{1.9}$$

Consequently, the current trial input is the sum of that used on the previous trial plus a correction term. In this particular case, the correction is based on the derivative of the previous trial error, and it is assumed that this term can be computed for measured signals without introducing unacceptable error. This law has the proportional plus derivative (PD) structure.

Error convergence in k, termed trial-to-trial error convergence, i.e.

$$\lim_{k\to\infty} \|e_k(t)\| = 0 \tag{1.10}$$

where $\|\cdot\|$ is any appropriate norm on the underlying function space, is a fundamental requirement in ILC. One condition for this property, see also Chapter 4, under the ILC law (1.9) applied to (1.8) is that

$$\|I - CBK_d\| < 1 \tag{1.11}$$

and $\|\cdot\|$ also denotes the induced operator norm. The immediate observations on this condition are that it does not depend on the system state matrix A, and it cannot be satisfied if $CB = 0$. Hence, this ILC law guarantees trial-to-trial error convergence for single-input single-output (SISO) systems of relative degree one. (Design for linear systems with relative degree greater than unity is considered in Section 7.3.) As in the standard linear systems case, it is possible to write down ILC equivalents of commonly used control laws. For example, a proportional plus integral plus derivative (PID) ILC law for (1.8) is

$$u_{k+1}(t) = u_k(t) + K_pe_k(t) + K_i\int e_k(\tau)d\tau + K_d\dot{e}_k(t) \tag{1.12}$$

A proportional, or P-type, ILC law is obtained by setting $K_i = 0$ and $K_d = 0$ in (1.12), a D-type law by setting $K_p = 0$ and $K_i = 0$, and a PD-type law by setting $K_i = 0$. The ILC law in [12] is D-type, and the following observations [270] on this law are relevant.

1. The right-hand side of (1.12) for D-type ILC has the property that the input $u_{k+1}(t)$ is used at time instance t and its directly produced result $\dot{x}_{k+1}(t)$ is then transmitted to the output derivative $\dot{y}_{k+1}(t) = C\dot{x}_{k+1}(t)$, also termed a causal pair [270]. These operations are algebraic and occur at the same instance t when the control input is applied, i.e. $u_{k+1}(t)$ and $\dot{y}_{k+1}(t)$ are algebraically related.
2. To implement D-type ILC, the highest-order derivative signals of the system are required and, hence, potential difficulties if these cannot be measured or are very noisy due to numerical differentiation. For example, many robots are equipped with position, but not velocity or acceleration, sensors. Therefore, accelerations have to be obtained by differentiating the position measurements twice, and the result can be estimated with severe noise corruption. Hence, high-level noise on measurements can severely reduce the effectiveness of D-type ILC. (The design of stochastic ILC laws is considered in Chapter 12.)

Consider P-type ILC, i.e.

$$u_{k+1}(t) = u_k(t) + K_p e_k(t) \tag{1.13}$$

or

$$u_{k+1}(t) = u_k(t) + K_p(r(t) - y_k(t)) \tag{1.14}$$

Then $(y_k(t), u_k(t))$ on the right-hand side is not a causal pair [270] since the result of the input action applied on trial k is not produced on this trial. In particular, when $u_k(t)$ is applied to the system state-space model, its effects cannot be seen in the state $x_k(t)$ because of the system updating structure. Equivalently, the state $x_k(t)$ is the result of an input applied before t and not $u_k(t)$. Hence, this ILC law does not capture direction or trend information in the errors produced on previous trials. For example, if $r(t) - y_k(t) = 0$ the control law (1.14) stops learning but $\dot{r}(t) - \dot{y}_k(t)$ could be of any value. However, this form of control law does not require higher-order derivatives.

The combination of P- and D-type ILC, resulting in PD ILC, is known to be effective, and the considerations summarized above led [270], in the case of nonlinear system dynamics, to propose anticipatory ILC with the following features:

- Use a causal pair of the action taken on the previous trial and use the result of this pair to compute the next trial input.
- Capture trend or direction information from the recorded previous trial errors but avoid using the highest-order derivatives present in the model.
- Keep the noise levels to a minimum to ease implementation.

Using the state-space model (1.8)

$$\begin{aligned} x_{k+1}(t+\Delta) &= x_{k+1}(t) + \int_t^{t+\Delta} \dot{x}_{k+1}(\tau)d\tau \\ &= x_{k+1}(t) + \int_t^{t+\Delta} (Ax_{k+1}(\tau) + Bu_{k+1}(\tau))d\tau \end{aligned} \tag{1.15}$$

and then $y_{k+1}(t+\Delta) = Cx_{k+1}(t+\Delta)$. Hence, $u_{k+1}(t)$ and $y_k(t+\Delta)$ form a causal pair in the sense of [270] with anticipatory action since $y_{k+1}(t+\Delta)$ is comparable to $\dot{y}_k(t)$ in capturing trend/directional information. This structure leads to the following form of ILC law:

$$u_{k+1}(t) = u_k(t) + L(r(t+\Delta) - y_k(t+\Delta)) \tag{1.16}$$

The design parameters are L and $\Delta > 0$, and this law is also known as phase-lead ILC.

Phase-lead ILC is not causal in the standard systems sense, where the general concept of time imposes a natural ordering into past, present, and future. Under this definition, $t + \Delta$ is in the future. The critical feature in ILC is that once a trial is complete, all information from it is available for use on the next trial at the cost of data storage. Still, this last aspect is application-specific, and the following is the definition of causality for ILC.

Definition 1.1 An ILC law is causal if and only if the input at time τ on trial $k+1$ is computed only from data that are available from trial $k+1$ in the time interval $0 \le t \le \tau$ (also written $t \in [0, \alpha]$) and from previous trials over the complete trial length $[0, \alpha]$.

In ILC, information from $t' > t$ can be used but only from previous trials. Control laws that use information from more than one previous trials to compute the current trial input are also possible, sometimes termed higher-order in trial [2] or simply higher-order iterative learning control (HOILC). Such control laws are considered again in Section 1.5 (and in Section 7.4).

The ILC laws discussed above do not necessarily require an accurate model of the system dynamics for implementation and design by tuning is quite common. Model-based control must be used for some applications, and this general area has been very heavily researched. One approach is to design the control law to minimize a suitably chosen cost function. The most common choice is the sum of quadratic terms in the state and control or, particular to ILC, the difference between the control signals applied on successive trials. Such cost functions, in an area known as norm optimal iterative learning control (NOILC), have been the subject of much research to derive algorithms that, in some cases, have been experimentally tested in the laboratory and then applied to physical systems. Optimal ILC is considered in Chapter 5.

1.4 ILC for Time-Varying Linear Systems

Consider linear time-varying (LTV) systems with input–output description

$$y(t) = g(t) + \int_0^t H(t, \tau) u(\tau) d\tau \tag{1.17}$$

where $u(t) \in \mathbb{R}^m$ and $y(t) \in \mathbb{R}^m$ are the input and output vectors, respectively, and the vector $g \in \mathbb{R}^m$ represents initial conditions and disturbances. Suppose that the reference trajectory vector $r(t)$ is given and also that the entries in $r(t), u(t), g(t)$, and $H(t, t')$ are continuously differentiable in t and t'. Then one ILC problem is described by

$$\begin{aligned} y_k(t) &= g(t) + \int_0^t H(t, t') u_k(t') dt' \\ e_k(t) &= r(t) - y_k(t) \\ u_{k+1}(t) &= u_k(t) + \Gamma(t) \dot{e}_k(t) \end{aligned} \tag{1.18}$$

where $\Gamma(t)$ is an $m \times m$ matrix in the PD ILC law to be designed. Also introduce the so-called λ norm

$$\|e_k\|_\lambda = \sup_{0 \le t \le \alpha} e^{-\lambda t} \max_{1 \le i \le m} |e_i(t)|, \quad \lambda > 0 \tag{1.19}$$

Then the following result is Theorem 1 in [12], where $\|\cdot\|_\infty$ is the matrix norm induced by the vector norm $\|f\| = \max_{1 \le i \le m} |f_i|$ and, hence, for an $m \times m$ matrix F

$$\|F\|_\infty = \max_{1 \le i \le m} \sum_{j=1}^m |f_{ij}| \tag{1.20}$$

Theorem 1.1 *Suppose that* $r(0) = g(0)$ *and*

$$\|I - H(t,t)\Gamma(t)\|_\infty < 1 \tag{1.21}$$

for all $0 \le t \le \alpha$ *and a given initial input* $u_0(t)$ *is continuous on* $0 \le t \le \alpha$. *Then there exists positive constants* λ *and* ρ_0 *such that*

$$\|\dot{e}_{k+1}\|_\lambda \le \rho_0 \|\dot{e}_k\|_\lambda \quad \text{and} \quad 0 \le \rho_0 < 1 \tag{1.22}$$

Moreover,

$$\|\dot{e}_k(t)\|_\lambda \to 0, \quad k \to \infty \tag{1.23}$$

and hence,

$$\dot{y}_k(t) \to \dot{r}(t), \quad k \to \infty \tag{1.24}$$

By the definition of the norm $\|\cdot\|_\lambda$, the convergence in both cases is uniform for $0 \le t \le \alpha$. Moreover, since $y_k(0) = g(0) = r(0)$, (1.24) implies that

$$y_k(t) \to r(t), \quad k \to \infty$$

uniformly for $0 \le t \le \alpha$.

Consider systems described by the state-space model (1.8), where the matrices are time-varying, i.e. A, B, and C are, respectively, replaced by $A(t), B(t)$, and $C(t)$. Then

$$y_{k+1}(t) = C(t)X(t)x_{k+1}(0) + \int_0^t C(t)X(t)X^{-1}(\tau)B(\tau)u_{k+1}(\tau)d\tau \tag{1.25}$$

where $X(t)$ is the unique matrix solution of the homogenous matrix differential equation:

$$\dot{X}(t) = A(t)X(t), \quad X(0) = I \tag{1.26}$$

Also, the time-invariant case is recovered when

$$g(t) = C(t)X(t)x(0) \quad \text{and} \quad H(t,\tau) = C(t)X(t)X^{-1}(\tau)B(\tau)$$

Hence, if $B(t) = B$, $C(t) = C$ and $\det(CB) \neq 0$, there exists a constant matrix such that

$$\|I - H(t,t)\Gamma\|_\infty = \|I - CB\Gamma\|_\infty < 1 \tag{1.27}$$

and if this condition holds application of the ILC law ensures that $y_k(t) \to r(t)$ uniformly in $t \in [0, \alpha]$ as $k \to \infty$. More recent results on time-varying ILC are given in Chapter 8, with supporting experimental results.

The λ-norm (1.19) is extensively used in trial-to-trial error convergence analysis, especially for nonlinear systems. However, it is a sufficient but not a necessary condition for convergence in the norm, and consequently, it can be very conservative. This aspect is considered again for linear systems in Chapter 3.

With the emphasis on digital implementation, discrete-time ILC is considered next. The outcome will be another representation for the system dynamics that provides the setting for a substantial volume of research, with follow-through to implementations and an extension to nonlinear discrete dynamics.

1.5 Discrete Linear Dynamics

The counterpart to (1.8), with N denoting the number of samples along a trial, for discrete-time linear systems is

$$\begin{aligned} x_k(p+1) &= Ax_k(p) + Bu_k(p) \\ y_k(p) &= Cx_k(p), \quad x_k(0) = x_0, \quad 0 \le p \le N-1, \quad k \ge 0 \end{aligned} \tag{1.28}$$

where the state, input, and output vectors are as for the differential case (1.8) with t replaced by p, and it is assumed that the state initial conditions are the same on each trial (where this assumption can be relaxed). Also, let q denote the forward shift-time operator defined by $qx(p) = x(p+1)$. Then an operator representation for the dynamics is

$$y_k(p) = G(q)u_k(p) + d(p) \tag{1.29}$$

and $d(p)$ is an exogenous signal that repeats on each trial. (Equivalently, the initial conditions are the same on each trial and there are no external disturbances.) Also,

$$\begin{aligned} y_k(p) &= C(qI - A)^{-1}Bu_k(p) + d(p) \\ d(p) &= CA^p x(0) \end{aligned} \tag{1.30}$$

The system $G(q)$ in (1.29) is a proper rational function of q and, in general, has a delay, or equivalently, a relative degree of h and is sufficiently general to capture infinite impulse response (IIR) and finite impulse response (FIR) systems. Assume that the system is stable, i.e. all eigenvalues of A have modulus strictly less than unity. The case when this property does not hold is considered again later in this section. Repeating disturbances, repeated nonzero initial conditions, and systems augmented with feedback and feedforward control can be included in the term $d(p)$ in (1.29).

Figure 1.4 illustrates the two-dimensional, or 2D, systems nature of ILC, information propagation from trial-to-trial (k) and along a trial (p). Control systems analysis for 2D systems is well developed in theory, and ILC provides an application area discussed in the next section. Let $r(p)$ denote the reference trajectory, and then $e_k(p) = r(p) - y_k(p)$ is the error on trial k. An extensively used ILC

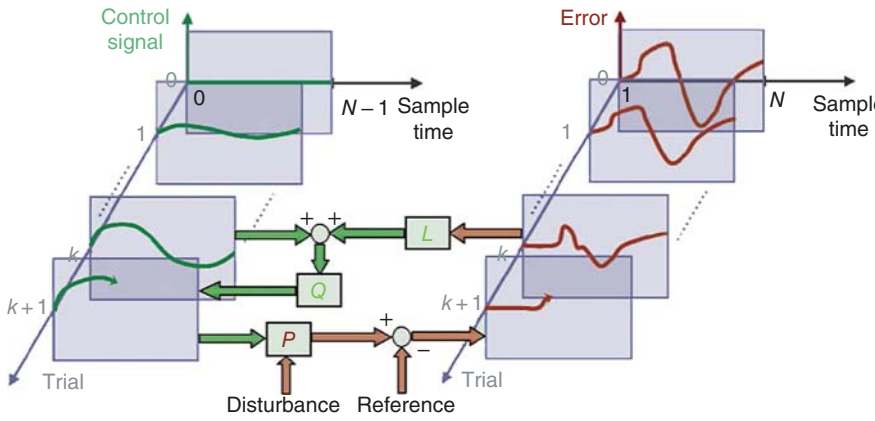


Figure 1.4 Illustrating the 2D structure of ILC.

law, stated for SISO systems with an immediate generalization to multiple-input multiple-output (MIMO) systems, is

$$u_{k+1}(p) = Q(q)(u_k(p) + L(q)e_k(p+1)) \tag{1.31}$$

where $Q(p)$ is termed the Q-filter and $L(p)$ the learning function. There are many variations of (1.31), including time-varying, nonlinear functions, and trial-varying functions. An application where trial-varying ILC has a role is in smart rotors for wind turbines considered in Section 9.1.2. Current trial feedback is a method of applying feedback action in conjunction with ILC and in this case (1.31) is extended to

$$u_{k+1}(p) = Q(q)(u_k(p) + L(q)e_k(p+1)) + C(q)e_{k+1}(p) \tag{1.32}$$

The current error term $C(q)e_{k+1}(p)$ is feedback action on the current trial.

Write (1.32) as

$$u_{k+1}(p) = w_{k+1}(p) + C(q)e_{k+1}(p) \tag{1.33}$$

where, by routine algebraic manipulations,

$$w_{k+1}(p) = Q(q)\left[w_k(p) + (L(q) + q^{-1}C(q))e_k(p+1)\right] \tag{1.34}$$

and the feedforward part of current trial ILC is identical to (1.31) with learning function $L(q) + q^{-1}C(q)$. Consequently, the ILC law (1.31) with learning function $L(q) + q^{-1}C(q)$ combined with a feedback controller in a parallel architecture is equivalent to the complete current trial ILC.

Expanding $G(q)$ in (1.29), with $h = 1$ for simplicity, as an infinite power series gives

$$G(q) = g_1 q^{-1} + g_2 q^{-2} + g_3 q^{-3} + \cdots \tag{1.35}$$

where the g_i are the Markov parameters. Moreover, the g_i forms the impulse response and since $CB \neq 0$ is assumed, $g_1 \neq 0$ and from (1.28), $g_j = CA^{j-1}B$. Introduce the vectors

$$Y_k = \begin{bmatrix} y_k(1) \\ y_k(2) \\ \vdots \\ y_k(N) \end{bmatrix}, \quad U_k = \begin{bmatrix} u_k(0) \\ u_k(1) \\ \vdots \\ u_k(N-1) \end{bmatrix}, \quad d = \begin{bmatrix} d(1) \\ d(2) \\ \vdots \\ d(N) \end{bmatrix} \tag{1.36}$$

and, hence, the dynamics can be written as follows:

$$Y_k = GU_k + d \tag{1.37}$$

$$G = \begin{bmatrix} g_1 & 0 & \dots & 0 \\ g_2 & g_1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_N & g_{N-1} & \cdots & g_1 \end{bmatrix} \tag{1.38}$$

The entries in Y_k and d are shifted by one-time step. Moreover, the relative degree is unity, to account for the one-step delay in the system, which ensures that G is invertible. If there is an $h > 1$ step delay or, equivalently, the first nonzero Markov parameter is $CA^{h-1}B$, the lifted system representation is

$$\begin{bmatrix} y_k(h) \\ y_k(h+1) \\ \vdots \\ y_k(h+N-1) \end{bmatrix} = G_h \begin{bmatrix} u_k(0) \\ u_k(1) \\ \vdots \\ u_k(N-1) \end{bmatrix} + \begin{bmatrix} d(h) \\ d(h+1) \\ \vdots \\ d(h+N-1) \end{bmatrix} \tag{1.39}$$

$$G_h = \begin{bmatrix} g_h & 0 & \dots & 0 \\ g_{h+1} & g_h & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_{h+N-1} & g_{h+N-2} & \cdots & g_h \end{bmatrix} \tag{1.40}$$

In the lifted form, the time and trial domain dynamics are replaced by an algebraic updating in the trial index. Consequently, the along the trial dynamics are absorbed into the super-vector, and simultaneous consideration of along the trial dynamics and trial-to-trial error convergence is not possible. In the 2D approach considered in the next section, such simultaneous consideration is possible.

The ILC law (1.31) can also be written in the lifted form, and the Q-filter and learning function L can be noncausal functions of the impulse response

$$\begin{aligned} Q(q) &= \cdots + q_{-2}q^2 + q_{-1}q + q_0 + q_1 q^{-1} + q_2 q^{-2} + \cdots \\ L(q) &= \cdots + l_{-2}q^2 + l_{-1}q + l_0 + l_1 q^{-1} + l_2 q^{-2} + \cdots \end{aligned} \tag{1.41}$$

and hence,

$$U_{k+1} = QU_k + LE_k \tag{1.42}$$

$$E_k = R - Y_k \tag{1.43}$$

$$R = \begin{bmatrix} r(1) & r(2) & \dots & r(N) \end{bmatrix}^T \tag{1.44}$$

$$Q = \begin{bmatrix} q_0 & q_{-1} & \cdots & q_{-(N-1)} \\ q_1 & q_0 & \cdots & q_{-(N-2)} \\ \vdots & \vdots & \ddots & \vdots \\ q_{N-1} & q_{N-2} & \cdots & q_0 \end{bmatrix} \tag{1.45}$$

$$L = \begin{bmatrix} l_0 & l_{-1} & \dots & l_{-(N-1)} \\ l_1 & l_0 & \dots & l_{-(N-2)} \\ \vdots & \vdots & \ddots & \vdots \\ l_{N-1} & l_{N-2} & \cdots & l_0 \end{bmatrix} \tag{1.46}$$

If $Q(q)$ and $L(q)$ are causal functions, then

$$q_{-1} = q_{-2} = \cdots = 0, \quad l_{-1} = l_{-2} = \cdots = 0 \tag{1.47}$$

and the matrices Q and L are lower triangular. The matrices G, Q, and L are also Toeplitz, i.e. all entries along each diagonal are equal. This setting also extends to LTV systems, but the corresponding matrices do not have the Toeplitz structure. Next, the z transform description is introduced.

The one-sided z-transform of a signal $x(j),\ j = 0,1, \dots,$ is

$$x(z) = \sum_{j=0}^{\infty} x(j) z^{-j} \tag{1.48}$$

and the frequency response is defined by $z = e^{j\theta},\ \theta \in [-\pi, \pi]$. To apply the z transform, it is necessary to assume $N = \infty$, even though the trial length is finite, and the implications of this difference will arise at various places throughout this book (next in Section 3.1).

In z-transform terms, the system and control law dynamics are described by

$$y_k(z) = G(z)u_k(z) + D(z) \tag{1.49}$$

$$u_{k+1}(z) = Q(z)(u_k(z) + zL(z)e_k(z)) \tag{1.50}$$

$$e_k(z) = r(z) - y_k(z) \tag{1.51}$$

The z term in this last equation emphasizes the forward time shift, and for an h time-step delay, z^h replaces z and the definition of causality given in Definition 1.1 for differential dynamics has an obvious discrete counterpart not formally stated.

Unlike the standard concept of causality, a noncausal ILC law is implementable in practice because the entire time sequence of data is available from all previous trials. Consider the noncausal ILC law

$$u_{k+1}(p) = u_k(p) + k_p e_k(p+1) \tag{1.52}$$

and the causal ILC law

$$u_{k+1}(p) = u_k(p) + k_p e_k(p) \tag{1.53}$$

Then a disturbance $d(p)$ contributes to the error as

$$e_k(p) = r(p) - G(q)u_k(p) - d(p) \tag{1.54}$$

Hence, noncausal ILC anticipates the disturbance $d(p+1)$ and compensates with the control action $u_{k+1}(p)$. The causal ILC law has no anticipation since $u_{k+1}(p)$ compensates for the disturbance $d(p)$ with the same time index p. Causality also has consequences for feedback equivalence where the final, or converged, control, denoted u_∞, can instead be obtained by a feedback controller. It can be shown [92] that there is a feedback equivalence for causal ILC laws, and the equivalent controller can be obtained directly from the ILC law.

The assertion in [92] is "causal ILC laws are of limited (or no) use since the same control action can be obtained by applying the equivalent feedback controller without the learning process." There are, however, critical limitations [33, 193] to this equivalence, and these are as follows:

- A noise-free requirement must be assumed.
- As the ILC performance increases, the gain of the equivalent feedback controller also increases. In the presence of noise, a high gain can lead to performance degradation and equipment damage. Hence, casual ILC laws are still of interest, and this equivalence was already known in the repetitive process/2D systems literature. See the next section.
- The equivalent feedback controller may not be stable.
- There is no equivalence for noncausal ILC as a feedback controller reacts to errors.

In [176] the system of Figure 1.5 with four inputs is considered, i.e. the reference signal $r(p)$, an externally generated control signal $u(p)$ and $w(p)$ and $v(p)$, respectively, denote load and measurement disturbances. The measured output is $y(p)$ and the controlled variable is $z(p)$. This system can also have an internal feedback and hence, the blocks G_u, G_r, G_w, and G_v contain the system to be controlled in addition to the controlled dynamics.

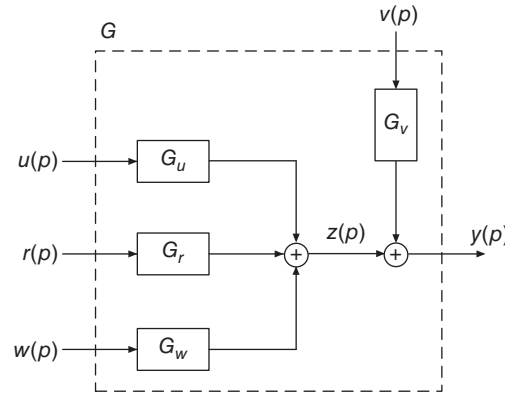
The equations describing the dynamics of this general representation in the SISO case, with an immediate generalization to MIMO systems, are given in lifted form by a system of the form

$$\begin{aligned} Z_k &= G_{r,k}R + G_{u,k}U_k + G_{w,k}W_k \\ Y_k &= Z_k + G_{v,k}V_k \end{aligned} \tag{1.55}$$

where the notation is as above, and W_k, V_k are the lifted vectors corresponding to $v_k(p)$, $w_k(p)$ and

$$Z_k = \begin{bmatrix} z_k(0) & \dots & z_k(N-1) \end{bmatrix}^T \tag{1.56}$$

Figure 1.5 ILC in general form.



This representation includes trial-variant and time-variant systems. If the system is both time- and trial-invariant, the dynamics can be described in the operator setting as

$$z_k(p) = G_r(q)r(p) + G_u(q)u_k(p) + G_w(q)w_k(p)$$
$$y_k(p) = z_k(p) + G_v v_k(p) \tag{1.57}$$

and this representation can be used for frequency-domain analysis. If time-varying dynamics is written in lifted form, some of the structural properties of the corresponding matrices in the time-invariant case are not preserved.

A lifted model can also be constructed for HOILC where, as one example, consider the following law from [176], which uses data from the previous $M > 1$ trials:

$$u_{k+1}(p) = \sum_{j=k-M+1}^{k} \left(Q_{k-j+1}(u_j(p) + L_{k-j+1}e_j(p)) \right) \tag{1.58}$$

Then

$$u_{k+1}(p) = \sum_{j=k-M+1}^{k} Q_{k-j+1}(I - L_{k-j+1}G_u)u_j(p) + \sum_{j=1}^{M} Q_j L_j (I - G_r)r(p) \tag{1.59}$$

or, on defining,

$$U_k = \begin{bmatrix} u_k^T(p) & u_{k-1}^T(p) & \dots & u_{k-M+1}^T(p) \end{bmatrix}^T \tag{1.60}$$

(1.59) can be written as follows:

$$u_{k+1} = \begin{bmatrix} H_1 & H_2 & \dots & H_M \\ I & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & I & 0 \end{bmatrix} u_k + \begin{bmatrix} F \\ 0 \\ \vdots \\ 0 \end{bmatrix} R \tag{1.61}$$

with $H_j = Q_j(I - L_j G_u)$ and $F = \sum_{j=1}^{M} Q_j L_j (I - G_r)$.

A HOILC law for differential dynamics can be found in [50]. These forms of HOILC laws can also include anticipatory action. An obvious question for investigation is when does HOILC bring advantages? Further coverage of HOILC is given in Chapters 3 and 7.

1.6 ILC in a 2D Linear Systems/Repetitive Processes Setting

1.6.1 2D Discrete Linear Systems and ILC

Multidimensional, or nD, systems originally arose from the need to have a mathematical setting to address problems whose formulation and solution required the use of functions and polynomials in more than one complex or real variable. Recently emerging areas where nD systems theory has been used include grid sensor networks and evidence filtering. In particular, wireless sensor networks consist of large numbers of resource-constrained, embedded sensor nodes and are an emerging candidate for many distributed applications. Some of these applications require regularly placed nodes in a spatial grid, often sampling the sensors periodically over time. Agriculture and environmental monitoring applications often favor a grid or mesh topology. Spatially distributed sensor lattices are also essential in surveillance, target location, and tracking applications. In [229, 230], several case studies on the application of nD control systems design are given, where for ILC, the case of interest is when $n = 2$..

Many models are used to describe nD linear systems, and one of the two heavily studied from a control systems standpoint is the Fornasini–Marchesini model [71]. An alternative model that describes how a dynamic process evolves over the 2D plane is the Roesser state-space model [227] where a state vector is defined for each direction of information propagation. If these vectors are denoted by $x^h(n_1, n_2)$, and $x^v(n_1, n_2)$, respectively, the state-space model is

$$\begin{bmatrix} x^h(n_1+1, n_2) \\ x^v(n_1, n_2+1) \end{bmatrix} = \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix} \begin{bmatrix} x^h(n_1, n_2) \\ x^v(n_1, n_2) \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u(n_1, n_2)$$
$$y(n_1, n_2) = \begin{bmatrix} C_1 & C_2 \end{bmatrix} \begin{bmatrix} x^h(n_1, n_2) \\ x^v(n_1, n_2) \end{bmatrix} \tag{1.62}$$

The links between ILC and 2D linear systems are evident, information propagation from trial-to-trial (k) and along the trial (p). Moreover, the first reported work [136] on the use of the 2D systems setting for ILC analysis and design used the Roesser model. Consider the state-space model (1.28) and

$$u_{k+1}(p) = u_k(p) + \Delta u_k(p) \tag{1.63}$$

where the control applied on the next trial is the sum of that used on the previous trial plus a correction denoted by $\Delta u_k(p)$. Using $e_k(p) = r(p) - y_k(p)$ it follows that

$$e_{k+1}(p) - e_k(p) = -CA\eta_k(p) - CB\Delta u_k(p-1) \tag{1.64}$$

where

$$\eta_k(p) = x_{k+1}(p-1) - x_k(p-1)$$

or, using (1.63) and the state equation in (1.28),

$$\eta_k(p+1) = A\eta_k(p) + B\Delta u_k(p-1) \tag{1.65}$$

Selecting

$$\Delta u_k(p) = Ke_k(p+1) \tag{1.66}$$

results in controlled systems dynamics described by the following 2D Roesser state-space model

$$\begin{bmatrix} \eta_k(p+1) \\ e_{k+1}(p) \end{bmatrix} = \begin{bmatrix} A & BK \\ -CA & I - CBK \end{bmatrix} \begin{bmatrix} \eta_k(p) \\ e_k(p) \end{bmatrix} \tag{1.67}$$

Hence, 2D systems theory for the Roesser state-space model, or that for the Fornasini–Marchesini state-space model, can be applied to ILC analysis and design.

One immediate conclusion from (1.67) is that the system-state matrix A and, therefore, the dynamics along any trial are independent of this control law. Hence, if A is an unstable matrix or along the trial dynamics have unacceptable transients, then one option is to design a preliminary feedback loop to remove these difficulties, resulting in a two-stage control law design.

The general 2D systems approach to ILC analysis and design, see Chapter 7, provide a means of avoiding this resulting two-stage design. For example, consider the ILC law (1.63) with

$$\Delta u_k(p) = -K_1\eta_k(p+1) + K_2 e_k(p+1) \tag{1.68}$$

where K_1 and K_2 are compatibly dimensioned matrices to be selected.

Then the controlled dynamics are described by the following Roesser state-space model:

$$\begin{bmatrix} \eta_k(p+1) \\ e_{k+1}(p) \end{bmatrix} = \begin{bmatrix} A - BK_1 & BK_2 \\ -CA + CBK_1 & I - CBK_2 \end{bmatrix} \begin{bmatrix} \eta_k(p) \\ e_k(p) \end{bmatrix} \tag{1.69}$$

and it is possible to design K_1 and K_2 simultaneously in this setting to control trial-to-trial error convergence and the along the trial dynamics.

1.6.2 ILC in a Repetitive Process Setting

Many processes make repeated completions of the same task or operation over a finite interval. Upon completing each, the process is reset to the initial location, ready for the next execution. Call each completion a pass, and the output produced the pass profile. In [70] the term "multi-pass process," subsequently changed to repetitive process, was introduced to describe the operation of longwall coal cutting. The novel feature is that the pass profile produced on the previous pass contributes to the next pass profile and hence has the ILC structure. Thus, as in most of the ILC literature, pass and pass length, respectively, are replaced by trial and trial length from this point onward.

Variables in a repetitive process model have to be described by a variable denoting the trial number and another for the dynamics produced on any trial. Moreover, any trial dynamics can be functions of differential or discrete variables, where the latter can arise from sampling of the former. In this book, the notation used for differential dynamics is of the form $h_k(t),\ 0 \le t \le \alpha, k \ge 0$, where h is the vector or scalar-valued variable under consideration, $\alpha < \infty$ denotes the trial length and k denotes the trial number. For discrete dynamics, the notation is of the form $h_k(p),\ 0 \le p \le N-1 < \infty, k \ge 0$, where N denotes the number of samples along the trial (N times the sampling period gives the trial length α).

Differential linear repetitive processes are described by the state-space model [228]

$$\begin{aligned} \dot{x}_{k+1}(t) &= Ax_{k+1}(t) + Bu_{k+1}(t) + B_0 y_k(t) \\ y_{k+1}(t) &= Cx_{k+1}(t) + Du_{k+1}(t) + D_0 y_k(t) \end{aligned} \tag{1.70}$$

where on trial k, $x_k(t) \in \mathbb{R}^n$ is the state vector, $y_k(t) \in \mathbb{R}^m$ is the trial profile vector, and $u_k(t) \in \mathbb{R}^l$ is the vector of control inputs. The simplest possible form of boundary conditions for these processes is

$$\begin{aligned} x_{k+1}(0) &= d_{k+1}, \quad k \ge 0 \\ y_0(t) &= f(t), \quad 0 \le t \le \alpha \end{aligned} \tag{1.71}$$

where the vector d_{k+1} has constant entries and the entries in $f(t)$ are known functions of t over the trial length.

Discrete linear repetitive processes are described by the state-space model

$$\begin{aligned} x_{k+1}(p+1) &= Ax_{k+1}(p) + Bu_{k+1}(p) + B_0 y_k(p) \\ y_{k+1}(p) &= Cx_{k+1}(p) + Du_{k+1}(p) + D_0 y_k(p) \end{aligned} \tag{1.72}$$

where on trial k, $x_k(p), y_k(p)$, and $u_k(p)$ are of the form of (1.70) with t replaced by p. Also, the boundary conditions are the discrete equivalent of (1.71).

The essential difference between discrete linear repetitive processes and 2D linear systems described by the Roesser and Fornasini–Marchesini state-space models is that information propagation in one of the two separate directions, along the trial, only occurs over the fixed-finite duration of the trial length. Therefore, as shown in Chapter 7, the repetitive process setting has more advantages than the Roesser and Fornasini–Marchesini state-space models for ILC analysis. Moreover, repetitive process-based ILC designs have seen experimental application, unlike those in the Roesser or Fornasini–Marchesini state-space model setting.

Remark 1.1 Other forms of boundary conditions for repetitive processes can be defined, see, e.g. [228] but are not required in this book. No explicit mention of the boundary conditions for linear repetitive processes will be made from this point onward.

1.7 ILC for Nonlinear Dynamics

The original work on ILC for nonlinear dynamics was for robotics where [12] considered an n-degree of freedom serial link robot whose dynamics, adopting the notation in [12], are described by the general form

$$R(\theta(t))\ddot{\theta}(t) + f(\theta(t), \dot{\theta}(t)) + g(\theta(t)) = \tau(t) \tag{1.73}$$

where

$$\theta(t) = \begin{bmatrix} \theta_1(t) & \theta_2(t) & \dots & \theta_n(t) \end{bmatrix}^T \tag{1.74}$$

is the joint angles coordinates vector and

$$\tau(t) = \begin{bmatrix} \tau_1(t) & \tau_2(t) & \dots & \tau_n(t) \end{bmatrix}^T \tag{1.75}$$

is the generalized force vector, $R(\theta(t))$ is the inertia matrix and is usually positive-definite, $f(\theta(t), \dot{\theta}(t))$ consists of centrifugal and Coriolis forces and other nonlinear characteristics such as frictional forces and $g(\theta(t))$ is the potential energy.

Often in robotics, the task, or reference signal in ILC terminology, is described in Cartesian or other task-specific coordinates. In the current case, it is assumed, for simplicity, that a desired motion of the manipulator is determined from a description of the task as a time function $r(t)$, $0 \leq t \leq \alpha$, for the joint coordinates $\theta(t)$ and an extension to task-oriented coordinates is straightforward.

In [12], a local linear model approximation in the form of a time-varying state-space was used to design a PD ILC law. Of course, this analysis has limitations, and nonlinear ILC model-based analysis and design has received a substantial amount of attention in the literature, particularly trial-to-trial error convergence proofs using the λ-norm defined by (1.19). The role of this norm in ILC analysis will be considered in Chapter 3.