15th Triennial World Congress, Barcelona, Spain

HIGHER-ORDER ITERATIVE LEARNING CONTROL BY POLE PLACEMENT AND NOISE FILTERING

Minh Q. Phan
Dartmouth College, Hanover, NH 03755

Richard W. Longman
Columbia University, New York, NY 10027

Abstract: This paper makes a comprehensive examination of likely sources that may give rise to higher-order iterative learning control (ILC) laws. Possibilities considered are higher-order model structure, improved learning speed, minimization of various quadratic cost functions, ILC designs based on predictive control, pole placement, direct and indirect adaptive control, and noise filtering. It is shown that among these possibilities, only a non-standard case of pole placement and noise filtering will naturally result in iterative learning controllers with orders higher than one. *Copyright © 2002 IFAC.*

Keywords: Learning control, iterative improvement, Kalman filter, singular value decomposition.

1. INTRODUCTION

Iterative learning control (ILC) develops controllers that aim to produce zero output tracking error by repeating a task over and over many times. ILC is capable of eliminating the effects of unknown repeating disturbances on the system output automatically if such disturbances are present. After each repetition or trial, the system is reset and the learning controller makes a correction to the input to be used in the next trial. A summary of past and recent developments of the field can be found in Bien and Xu (1998). The vast majority of ILC laws in the literature are based on the tracking error and the control input of the previous trial alone. This is referred to as first-order ILC. Recently, there has been considerable interest in using information from a number of past repetitions to determine the current control input correction. This strategy has been referred to as higher-order ILC. For example, it was noted in Bien and Huh (1989) that the use of higher-order ILC could sometimes result in an improved rate of convergence over the first-order approach. However, Moore (1999) argued that the real value of higher-order ILC was to mitigate the effects of disturbance and noise. Recently, it was found in Phan, Longman, and Moore (2000) that certain higher-order ILC laws have their first-order equivalents. Thus the real value of higher-order ILC is still an open question.

Against this background, one fundamental question remains unanswered: Under what conditions do higher-order ILC automatically arise? To answer this question, we take advantage of a previously developed repetition-domain formulation of ILC, which allows the problem to be put in modern control form, so that typical modern control techniques can be applied to derive ILC laws. In this paper, a rather comprehensive set of such mechanisms is explored, including higher-order model structure, minimization of various quadratic cost functions, ILC designs based on predictive control, pole placement, indirect and direct adaptive control, and noise filtering. We will show that most of these possibilities do not result in higher-order ILC. Only a non-standard case of pole placement where the order of the controller is higher than the order of the system, and a case of noise filtering result in ILC laws with orders higher than one. This study helps explain why it has seemed difficult to justify the need for higher-order ILC in spite of the fact that the idea itself appears so natural.

2. REPETITION-DOMAIN REPRESENTATION

We now describe a repetition-domain representation first formulated in Phan and Longman (1988) that is particularly convenient for subsequent ILC design and analysis. Consider an n-th order single-input single-output system of the general form

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + d(k) \\ y(k) &= Cx(k) \end{aligned} \tag{1}$$

where A, B, C describe the known system dynamics. Extension to the multiple-input multiple-output case is straightforward. The index k denotes the time $t = k\Delta t$ where Δt is an appropriate sampling interval. The quantities $x(k)$, $y(k)$, $d(k)$, $u(k)$ denote the unknown system state, known output, unknown disturbance, and ILC input, respectively. Typically ILC is implemented in conjunction with an existing feedback controller. The role of ILC is then to provide a correction to the command to the control system. In this case the matrices A, B, C represent the closed-loop system with the embedded existing feedback controller.

The ILC objective is to make the system track a desired output trajectory $y^*(k)$, $k = 1,2,3,...,p$ by repeated trials of the control input time history $u(k)$ for $k = 0,1,2,...,p-1$. After each trial, the system returns to the same initial condition before the next trial commences. It is assumed that the unknown disturbance $d(k)$ is the same from one repetition (or trial) to the next. Using the subscript j to denote a trial or repetition number, the relationship between an input time history and the resultant output time history at any repetition j can be written as

$$\underline{y}_j = P\underline{u}_j + \underline{w} \tag{2}$$

where

$$\underline{y}_j = \begin{bmatrix} y_j(1) \\ y_j(2) \\ y_j(3) \\ \vdots \\ y_j(p) \end{bmatrix}, \quad \underline{u}_j = \begin{bmatrix} u_j(0) \\ u_j(1) \\ u_j(2) \\ \vdots \\ u_j(p-1) \end{bmatrix}, \quad \underline{w} = \begin{bmatrix} w(0) \\ w(1) \\ w(2) \\ \vdots \\ w(p-1) \end{bmatrix}$$

$$P = \begin{bmatrix} CB & & & & \\ CAB & CB & & & \\ CA^2B & CAB & \ddots & & \\ \vdots & \ddots & \ddots & CB & \\ CA^{p-1}B & \cdots & CA^2B & CAB & CB \end{bmatrix} \tag{3}$$

The disturbance vector $\underline{w}$ incorporates the effect of the initial state $x(0)$, and the repeating disturbance $d(k)$ in the time domain. To eliminate the effect of $\underline{w}$, define a backward difference operator δ_j in the repetition domain variable j, applied to any variable v to be $\delta_j v = v_j - v_{j-1}$. Applying this operator to (2) yields

$$\delta_j \underline{y} = P\delta_j \underline{u} \tag{4}$$

With the tracking error defined as $\underline{e}_j = \underline{y}^* - \underline{y}_j$, it follows that $\delta_j \underline{y} = -\delta_j \underline{e}$. Hence, the tracking error version of (4) is

$$\delta_j \underline{e} = -P\delta_j \underline{u} \tag{5}$$

Notice that the effects of any repeating initial condition and repeating disturbance are automatically eliminated by the backward difference operator. Consequently, a stable ILC law designed from (4) or (5) will automatically compensate for the unknown initial condition and disturbance.

In writing (2)-(5) we implicitly assume that $CB \neq 0$, i.e., the system time delay is one as is the case when discretizing a continuous-time system with a zero-order hold on the input. If additional time delays are present, say $CB = 0$ but $CAB \neq 0$ then one adjusts the definitions in (3) so that P again has non-zero elements on the main diagonal (CAB). When there is a direct transmission term D in the time domain model, one can specify the p-time step long desired trajectory at time steps $k = 0,1,2,...,p-1$ to be achieved by adjusting the inputs at these same time steps. In this case the definition for $\underline{y}_j$ can start from time step 0 instead of 1 and the P matrix starts with D rather than CB. With this adjustment, the equations again have the form of (2) and (4) or (5) with an invertible P.

3. ILC BASED ON HIGHER-ORDER MODEL

In standard feedback control theory, one possible reason for needing high-order control is the high order of the system model. We examine if this need arises in the case of ILC where higher orders mean orders higher than one. Notice that (2) is a static model. The dimension of P depends on the number of time steps in the trajectory, not on the order (the number of states) of the time-domain model (1). To eliminate the unknown repeating disturbance term, it is sufficient to use the difference operator δ_j to produce (4). Equations (4) can be viewed as a "state-space" model in the repetition domain,

$$\begin{aligned} \underline{z}_{j+1} &= I\underline{z}_j + P\delta_{j+1}\underline{u} \\ \underline{y}_j &= I\underline{z}_j \end{aligned} \tag{6}$$

where the "system" matrix is I, the learning input influence matrix is P, the output influence matrix is I, and the learning control variable is $\delta_{j+1}\underline{u}$. The fact that the repetition-domain system matrix is an identity matrix carries several implications. The system (6) is controllable if and only if P is square and full rank. Since the output matrix is I, (6) is automatically observable because the "state" is directly measured. For a standard state-space model of modern control, if there are fewer output measurements than the number of states, but the system is observable, then it is possible to convert a first-order state-space model into an equivalent higher-order input-output model. In the case of ILC, however, because the "state" is directly measured, the repetition-domain model is naturally in first-order form. Of course it is possible to create a repetition-domain model with order higher than one, but such a higher-order model is artificial in the sense that it merely contains cancelling pole and zero pairs corresponding to extra repetition-domain "observable" but "uncontrollable" dynamics. We therefore conclude that the need for higher-order ILC is *not* dictated by the need for a higher-order repetition-domain representation.

4. HIGHER-ORDER ILC AND SPEED OF LEARNING

Next we address the possibility of needing a higher-order ILC to improve the speed of learning. Again the fact that the repetition-domain system matrix is identity plays a role in the following consideration.

Let us consider the case of a state-space model *A,B,C,* of standard modern control theory,

$$\begin{aligned} y(k) &= y(0) + CBu(k-1) + CABu(k-2) \\ &\quad + CA^2Bu(k-3) + \cdots + CA^{k-1}Bu(0) \end{aligned} \tag{7}$$

The output at any time step k is a convolution sum of the pulse response time history (given by the time-domain Markov parameters CA^kB), and the control input time history. Translating this result to the repetition-domain model for ILC, we have

$$\begin{aligned} \underline{y}_j &= \underline{y}_0 + P\delta_j\underline{u} + P\delta_{j-1}\underline{u} + \cdots + P\delta_1\underline{u} \\ &= \underline{y}_0 + P\left(\delta_j\underline{u} + \delta_{j-1}\underline{u} + \cdots + \delta_1\underline{u}\right) \\ &= \underline{y}_0 + P\left(\underline{u}_j - \underline{u}_0\right) \end{aligned} \tag{8}$$

As long as $A \neq I$ it is clear from (7) that bringing the output to a desired value in one time step or a number of time steps can involve very different control effort. There can be a preference based on actuator saturation constraints. However, in ILC the repetition-domain system matrix is an identity matrix, hence in (8) all the repetition-domain Markov parameters are the same, equal to P. When the desired trajectory is feasible, then there is no issue of saturation (by definition), and the only requirement for achieving zero tracking error is that the sum of all corrections $\delta_j\underline{u}, \delta_{j-1}\underline{u}, .., \delta_1\underline{u}$ over all repetitions equals the change $\underline{u}_j - \underline{u}_0$ that could have been made in the first repetition to achieve zero error. The corrections can be done all at once or divided in any desired way over a number of repetitions, with the only requirement that the sum be the needed value. Thus, ILC can in theory achieve zero tracking error in one repetition using the first-order inverse ILC law

$$\delta_j\underline{u} = \underline{u}_j - \underline{u}_{j-1} = P^{-1}\underline{e}_{j-1} \tag{9}$$

Thus as far as speed of learning is concerned, a first-order inverse ILC controller gives the fastest learning possible. Often such an inverse controller is not desirable because it requires exact knowledge of P, and its sensitivity to noise in the system. Nevertheless we can still conclude that the benefit of using a higher-order ILC law is not in improved learning speed, but in other factors such as robustness to noise and modelling uncertainties. For example, in the next section we show a case where slowing down the learning rate is used by a first-order ILC law to avoid possible numerical ill-conditioning in the computation of P^{-1}.

5. ILC BASED ON QUADRATIC COST MINIMIZATION

In this section we investigate whether the use of various "typical" cost functions of optimal control applied to ILC problems can result in higher-order ILC laws.

Quadratic Cost 1: The simplest repetition-domain cost function that one may use has the form

$$J_j = \left[\underline{e}_j\right]^T Q\,\underline{e}_j + \left[\delta_j\underline{u}\right]^T R\,\delta_j\underline{u} \tag{10}$$

This cost function removes the numerical ill-conditioning of P^{-1} that would be required in an inverse ILC law such as (9). The learning speed is governed by varying the parameter R that penalizes large correction from one repetition to the next. Note that the effect of R goes away as the system converges to the desired inverse solution as $\delta_j\underline{u} \to 0$. Minimizing this cost function results in a first-order ILC law

$$\delta_j\underline{u} = \left(P^TQP + R\right)^{-1}P^TQ\,\underline{e}_{j-1} \tag{11}$$

Quadratic Cost 2: Next we consider a quadratic cost analogous to that used in standard optimal control theory, say for example, the infinite horizon case

$$J = \sum_{j=0}^{\infty}\left[\underline{e}_j\right]^T Q\,\underline{e}_j + \sum_{j=0}^{\infty}\left[\delta_{j+1}\underline{u}\right]^T R\,\delta_{j+1}\underline{u} \tag{12}$$

By direct analogy such an optimal ILC law is first-order with an optimal learning gain L^*,

$$\delta_j\underline{u} = L^*\,\underline{e}_{j-1} \tag{13}$$

where

$$L^* = \left(R + P^TSP\right)^{-1}P^TS \tag{14}$$

$$Q - SP\left(R + P^TSP\right)^{-1}P^TS = 0 \tag{15}$$

Quadratic Cost 3: Let us examine a cost function that involves future changes in the learning control correction as motivated by the concept of predictive control. The simplest form of receding-horizon predictive control is as follows. At each time step a sequence of control inputs that would bring the system state at some future time step to zero is computed, but only the first of that control sequence is used. The entire calculation is repeated at the next time step to generate the next control to use. The equivalent ILC version is to minimize

$$J_j = \underline{e}_{j+q-1}^T\underline{e}_{j+q-1} \tag{16}$$

where the q-repetition ahead future tracking error is a function of the current tracking error and future learning control correction

$$\underline{e}_{j+q-1} = \underline{e}_{j-1} - \left[P,\ P,\ \ldots,\ P\right]\begin{bmatrix} \delta_j\underline{u} \\ \delta_{j+1}\underline{u} \\ \vdots \\ \delta_{j+q-1}\underline{u} \end{bmatrix} \tag{17}$$

Minimizing (16) yields

$$\begin{bmatrix} \delta_j \underline{u} \\ \delta_{j+1} \underline{u} \\ \vdots \\ \delta_{j+q-1} \underline{u} \end{bmatrix} = \begin{bmatrix} P, P, \ ..., \ P \end{bmatrix}^{+} \underline{e}_{j-1} = \frac{1}{q} \begin{bmatrix} P^{-1} \\ P^{-1} \\ \vdots \\ P^{-1} \end{bmatrix} \underline{e}_{j-1} \quad (18)$$

where the plus (+) sign denotes the pseudo-inverse operation. From (18) the q-repetition ahead predictive ILC law is given by

$$\delta_j \underline{u} = \left(\frac{1}{q} \right) P^{-1} \underline{e}_{j-1} \quad (19)$$

Thus the effect of using predictive control for ILC looking q repetitions into the future is the same as using the first-order inverse control law (9), but making $1/q$ of the total correction needed each repetition. This is another important consequence of having the repetition-domain system matrix be an identity matrix.

Quadratic Cost 4: So far we have considered quadratic costs that involve the changes in the control instead of the control input directly. Our conclusions remain the same if the actual control input is involved instead. For example, minimizing

$$J_j = \left[\underline{e}_j \right]^T Q \underline{e}_j + \left[\underline{u}_j \right]^T R \underline{u}_j \quad (20)$$

results in the following ILC law

$$\delta_j \underline{u} = \left(P^T Q P + R \right)^{-1} \left[P^T Q \underline{e}_{j-1} - R \underline{u}_{j-1} \right] \quad (21)$$

Thus, in addition to the tracking error, the control input of the previous repetition is also involved in the ILC update. The resultant ILC law, however, still remains first-order. This case and the case of using ILC to arrive at the same optimal solution without initial knowledge of the plant were presented in Frueh and Phan (2000).

6. ILC BASED ON INDIRECT AND DIRECT ADAPTIVE CONTROL

One scenario where data from past repetitions affect the computation of the current ILC correction occurs when system identification is involved. In the ILC counterpart of indirect adaptive control, the Markov parameters that make up P of repetition-domain model (5) are identified, Phan and Longman (1989), Moore (1999). This approach results in a first-order ILC law with a repetition-varying learning gain. The update formula for the first-order learning gain involves data from two previous repetitions or trials. For simplicity, we illustrate the ILC counterpart of indirect adaptive control with the projection algorithm although more sophisticated identification algorithms can be used. The ILC law is

$$\delta_j \underline{u} = \hat{P}_{j-1}^{-1} \underline{e}_{j-1} \quad (22)$$

where

$$\hat{P}_{j-1} = \hat{P}_{j-2} - \frac{a \left[\delta_{j-1} \underline{e} + \hat{P}_{j-2} \delta_{j-1} \underline{u} \right] \left(\delta_{j-1} \underline{u} \right)^T}{c + \left(\delta_{j-1} \underline{u} \right)^T \delta_{j-1} \underline{u}} \quad (23)$$

and $c > 0, \ 0 < a < 2$. Notice that the update formula to estimate P_{j-1} involves the difference in the control and tracking error from two previous repetitions. For the ILC counterpart of direct adaptive control, the inverse of P can be directly identified by simply interchanging $\delta_{j-1} \underline{u}$ and $\delta_{j-1} \underline{e}$ in (23). The identified inverse can then be used as the ILC gain in (22).

7. GENERAL FORM OF LINEAR HIGHER-ORDER ILC

A linear n-th order ILC has the general form,

$$\begin{aligned} \delta_j \underline{u} = {} & L_1 \underline{e}_{j-1} + L_2 \underline{e}_{j-2} + \cdots + L_n \underline{e}_{j-n} \\ & - M_1 \delta_{j-1} \underline{u} - M_2 \delta_{j-2} \underline{u} - \cdots - M_{n-1} \delta_{j-n+1} \underline{u} \end{aligned} \quad (24)$$

Using (5), the dynamics of the ILC process in the repetition domain is given by

$$\begin{aligned} & \underline{e}_j - \left(I - P L_1 \right) \underline{e}_{j-1} + P L_2 \underline{e}_{j-2} + \cdots + P L_n \underline{e}_{j-n} \\ & = P M_1 \delta_{j-1} \underline{u} + P M_2 \delta_{j-2} \underline{u} + \cdots + P M_{n-1} \delta_{j-n+1} \underline{u} \end{aligned} \quad (25)$$

In the limit as $j \to \infty$, we desire $\underline{e}_j \to 0$ as $\delta_j \underline{u} \to 0$ (i.e., $\underline{u}_j$ converges to the necessary control input) by proper choice of the learning control gains in (24). Equation (24) represents a set of coupled and high dimensional difference equations. Each learning gain matrix has dimensions $p \times p$, where p is the number of time steps in the input and output trajectories. Designing the learning control gains is therefore not trivial. In the following we use the singular value decomposition (SVD) to reduce (25) to a set of uncoupled scalar equations so that the desired repetition-domain poles and zeros can be placed.

8. UNCOUPLING THE REPETITION MODEL BY THE SVD

In the following we work with (5) which describes how a change in the learning control input affects the tracking error in the subsequent repetition. The SVD of P is $P = USV^T$, which allows us to re-write (5) as

$$\delta_j \underline{e} = -USV^T \delta_j \underline{u} \quad (26)$$

Pre-multiplying both sides of (26) by the transpose of U yields

$$U^T \delta_j \underline{e} = -SV^T \delta_j \underline{u} \quad (27)$$

Recognizing $U^T \delta_j \underline{e} = \delta_j \left(U^T \underline{e} \right)$, $V^T \delta_j \underline{u} = \delta_j \left(V^T \underline{u} \right)$, $UU^T = U^T U = I$, $VV^T = V^T V = I$, define

$$\underline{\alpha}_j = U^T \underline{e}_j \ , \ \underline{\beta}_j = V^T \underline{u}_j \ , \ \underline{e}_j = U \underline{\alpha}_j \ , \ \underline{u}_j = V \underline{\beta}_j \quad (28)$$

so that (5) can be simplified further as

$$\delta_j \underline{\alpha} = -S \delta_j \underline{\beta} \quad (29)$$

where S is a diagonal matrix of the singular values $\sigma(k) > 0$, which can be arranged in descending order if desired,

$$S = \begin{bmatrix} \sigma(1) & & & \\ & \sigma(2) & & \\ & & \ddots & \\ & & & \sigma(p) \end{bmatrix} \quad \underline{\alpha} = \begin{bmatrix} \alpha(1) \\ \alpha(2) \\ \vdots \\ \alpha(p) \end{bmatrix} \quad \underline{\beta} = \begin{bmatrix} \beta(1) \\ \beta(2) \\ \vdots \\ \beta(p) \end{bmatrix} \quad (30)$$

The elements of $\underline{\alpha}$ (and $\underline{\beta}$) are the coefficients of the tracking error (and control input) time history on the orthonormal left (and right) singular vectors of P. Equations (28) give the one-to-one mappings between $\underline{\alpha}$ and $\underline{e}$, between $\underline{\beta}$ and $\underline{u}$ at any repetition j. Because S is diagonal, the elements of $\underline{\alpha}$ and $\underline{\beta}$ are uncoupled from each other,

$$\alpha_j(k) = \alpha_{j-1}(k) - \sigma(k) \delta_j \beta(k) \quad k = 1, 2, ..., p \quad (31)$$

9. HIGHER-ORDER ILC BY POLE PLACEMENT

We now illustrate how (31) can be used to design various iterative learning controllers. For example, let us consider a second-order ILC law

$$\beta_j(k) = \beta_{j-1}(k) + \delta_j \beta(k), \quad k = 1, 2, ..., p \quad (32)$$

where

$$\begin{aligned} \delta_j \beta(k) = {} & g_1(k) \alpha_{j-1}(k) + g_2(k) \alpha_{j-2}(k) \\ & - h_1(k) \delta_{j-1} \beta(k) \end{aligned} \quad (33)$$

The above ILC law is second-order because the current learning control correction is based on data from two previous repetitions. In general the scalar gains $g_i(k)$ are allowed to vary with k. Recall that the equations in (28) are used to transform back and forth between the tracking error and the control input time histories and their corresponding coefficients at any repetition j during the implementation of such an ILC law.

The stability of the learning process is governed by

$$\begin{aligned} \alpha_j(k) = \alpha_{j-1}(k) - \sigma(k) \big[& g_1(k) \alpha_{j-1}(k) + g_2(k) \alpha_{j-2}(k) \\ & - h_1(k) \delta_{j-1} \beta(k) \big] \end{aligned} \quad (34)$$

Equation (34) can be re-written as

$$\begin{aligned} \alpha_j(k) + \big[\sigma(k) g_1(k) - 1 \big] \alpha_{j-1}(k) + \big[\sigma(k) g_2(k) \big] \alpha_{j-2}(k) \\ = \big[\sigma(k) h_1(k) \big] \delta_{j-1} \beta(k) \end{aligned} \quad (35)$$

For pole placement ILC design, the scalar learning gains $g_i(k)$ and $h_i(k)$ can be found that place repetition domain poles of (35) at any desired locations. Note that this non-standard case of pole placement is not the common type of pole placement where the existing system open-loop poles are moved to new closed-loop locations. Instead, new poles (and zeros) are created and placed. Perhaps the closest analogy to standard feedback controller design occurs when the additional poles and zeros are introduced to cancel multiple harmonic disturbances.

10. HIGHER-ORDER ILC BY NOISE FILTERING

Consider the system (1) with process noise $\varepsilon_1(k)$ and measurement noise $\varepsilon_2(k)$ present,

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + d(k) + \varepsilon_1(k) \\ y(k) &= Cx(k) + \varepsilon_2(k) \end{aligned} \quad (36)$$

The repetition-domain representation is

$$\underline{e}_j = \underline{e}_{j-1} - P \delta_j \underline{u} + \underline{\varepsilon} \quad (37)$$

where $\underline{\varepsilon}$ incorporates the effect of process and measurement noise as seen by $\underline{e}_j$. The associated Kalman filter for (37) is

$$\hat{\underline{e}}_j = \hat{\underline{e}}_{j-1} - P \delta_j \underline{u} + K_{j-1} \left(\underline{e}_{j-1} - \hat{\underline{e}}_{j-1} \right) \quad (38)$$

Let the learning process be governed by an ILC law

$$\delta_j \underline{u} = L \underline{e}_{j-1} \quad (39)$$

with some suitably designed learning gain L such as that given in (11). First we examine the most obvious case where the tracking error provided by the above Kalman filter is used in place of the measured tracking error $\underline{e}_{j-1}$,

$$\begin{aligned} \delta_j \underline{u} &= L \hat{\underline{e}}_{j-1} \\ \hat{\underline{e}}_{j-1} &= \hat{\underline{e}}_{j-2} - P \delta_{j-1} \underline{u} + K_{j-2} \left(\underline{e}_{j-2} - \hat{\underline{e}}_{j-2} \right) \end{aligned} \quad (40)$$

Although this strategy is valid, closer examination of (40) reveals that the measured error $\underline{e}_{j-1}$, although available, is not used in the computation of $\delta_j \underline{u}$. This situation is different from the standard case of using an observer or Kalman filter in an observer-based state feedback control system. There the computation of $u(k)$, which occurs between time step $k-1$ and k, uses $y(k-1)$ which is the most recently measured output. In the ILC case, what we should use instead

is a true filter for $\underline{e}_{j-1}$ denoted by $\underline{e}^*_{j-1}$. From linear filtering theory, the optimal $\underline{e}^*_{j-1}$ can be back computed from the Kalman filtered $\underline{e}_j$ as

$$\begin{aligned}\underline{e}^*_{j-1} &= \hat{\underline{e}}_j + P\delta_j\underline{u} \\ &= \hat{\underline{e}}_{j-1} - P\delta_j\underline{u} + K_{j-1}\left(\underline{e}_{j-1} - \hat{\underline{e}}_{j-1}\right) + P\delta_j\underline{u} \\ &= \left(I - K_{j-1}\right)\hat{\underline{e}}_{j-1} + K_{j-1}\underline{e}_{j-1}\end{aligned} \tag{41}$$

Thus an ILC law that uses the error estimate that is in fact optimal because it uses all available data is

$$\delta_j\underline{u} = L\underline{e}^*_{j-1} \tag{42}$$

where $\underline{e}^*_{j-1}$ is given by (41) and the Kalman filtered error $\hat{\underline{e}}_{j-1}$ in (41) is given by (40).

Next we proceed to determine the order of the above learning control scheme. Adding and subtracting $L^{-1}\delta_j\underline{u}$ from (38) yields

$$\begin{aligned}\hat{\underline{e}}_j &= \left(I - K_{j-1}\right)\hat{\underline{e}}_{j-1} - P\delta_j\underline{u} + K_{j-1}\underline{e}_{j-1} \\ &\quad - L^{-1}\delta_j\underline{u} + L^{-1}\delta_j\underline{u} \\ &= \left[\left(I - K_{j-1}\right) - L^{-1}L\left(I - K_{j-1}\right)\right]\hat{\underline{e}}_{j-1} \\ &\quad + \left(K_{j-1} - L^{-1}LK_{j-1}\right)\underline{e}_{j-1} - \left(P - L^{-1}\right)\delta_j\underline{u} \\ &= -\left(P - L^{-1}\right)\delta_j\underline{u}\end{aligned} \tag{43}$$

Substituting (43) into (42) produces

$$\begin{aligned}\delta_j\underline{u} &= L\left(I - K_{j-1}\right)\hat{\underline{e}}_{j-1} + LK_{j-1}\underline{e}_{j-1} \\ &= -L\left(I - K_{j-1}\right)\left(P - L^{-1}\right)\delta_{j-1}\underline{u} + LK_{j-1}\underline{e}_{j-1}\end{aligned} \tag{44}$$

Thus, the above ILC law is second-order in $\delta_j\underline{u}$ but first-order in $\underline{e}_j$ as defined in (24).

11. CONCLUSION

Although the idea of higher-order iterative learning control seems natural, the current understanding of the topic is quite incomplete. Clearly it is possible that a well-designed higher-order ILC law will outperform a poorly designed first-order one. But the opposite is also possible. We believe that it is difficult to justify the merit of either design strategy through anecdotal evidence, and prefer to search for various ways in which higher-order ILC might naturally arise.

This paper performs an extensive search for possible sources for higher-order ILC. Using the repetition-domain formulation of ILC that puts the learning control problem in modern control format, a rather comprehensive set of standard controller design techniques were applied. Possible sources for higher-order ILC examined in this paper are higher-order model structure, improved learning speed, minimization of various typical quadratic cost functions, and design approaches based on predictive control, pole placement, direct and indirect adaptive control, and noise filtering. This paper showed that among these possibilities, only a non-standard case of pole placement where the order of the controller is higher than the order of the system, and a case of noise filtering naturally resulted in iterative learning controllers with orders higher than one. All others resulted in first-order ILC laws. The results of this paper help explain the prevalence of first-order ILC laws in the literature. They also reveal that the real need for higher-order ILC designs may not be as obvious as previously anticipated. However, incorporating a noise filter in the learning control scheme did produce an ILC law of second-order in the control correction but first-order in the tracking error. In this particular case the need for a higher-order ILC law over a first-order one was demonstrated. What we have not addressed in this paper is the issue of robustness to modelling error and their associated design approaches. Whether such a consideration would naturally lead to a higher-order ILC design remains a topic for future consideration.

ACKNOWLEDGMENT

The authors would like to thank Professor Kevin L. Moore for his insights on the topic of higher-order iterative learning control that prompted this work.

REFERENCES

Bien, Z. and Huh, K.M. (1989). Higher-Order Iterative Control Algorithm. *IEE Proceedings Part D, Control Theory and Applications*, **136**, 105-112.

Bien, Z. and Xu, J.-X. (1998). *Iterative Learning Control: Analysis, Design, Integration, and Applications,* Kluwer Academic Publishers, Boston, 1998.

Frueh, J.A. and Phan, M.Q. (2000). Linear Quadratic Optimal Learning Control (LQL). *International Journal of Control,* **73**, 832-839.

Moore, K.L. (1999). An Iterative Learning Control Algorithm for Systems with Measurement Noise. *Proceedings of the 38th IEEE Conf. on Decision and Control*, Phoenix, Arizona, pp. 270-275.

Phan, M.Q., Longman, R.W., and Moore, K.L. (2000). A Unified Formulation of Iterative Learning Control. *Advances in the Astronautical Sciences*, **105**, 93-111.

Phan, M.Q. and Longman, R.W. (1988). A Mathematical Theory of Learning Control for Linear Discrete Multivariable Systems. *Proceedings of the AIAA/AAS Astrodynamics Conference,* Minneapolis, Minnesota, pp. 740-746.

Phan, M.Q. and Longman, R.W. (1989). Indirect Learning Control with Guaranteed Stability. *Proceedings of the Johns Hopkins Conference on Information Sciences and Systems,* Baltimore, Maryland, pp. 125-131.