Iterative learning control and repetitive control for engineering practice

RICHARD W. LONGMAN†

This paper discusses linear iterative learning and repetitive control, presenting general purpose control laws with only a few parameters to tune. The method of tuning them is straightforward, making tuning easy for the practicing control engineer. The approach can then serve the same function for learning/repetitive control, as PID controllers do in classical control. Anytime one has a controller that is to perform the same tacking command repeatedly, one simply uses such a law to adjust the command given to an existing feedback controller and achieves a substantial decrease in tracking error. Experiments with the method show that decreases by a factor between 100 and 1000 in the RMS tracking error on a commercial robot, performing a high speed trajectory can easily be obtained in 8 to 12 trials for learning. It is shown that in engineering practice, the same design criteria apply to learning control as apply to repetitive control. Although the conditions for stability are very different for the two problems, one must impose a good transient condition, and once such a condition is formulated, it is likely to be the same for both learning and repetitive control.

1. Introduction

The concept of iterative learning control (ILC) suddenly began to flourish in 1984, motivated by robots doing repetitive tasks. Arimoto et al. (1984), Casalino and Bartolini (1984) and Craig (1984) are independent developments of similar ideas that year, with Uchiyama (1978) being one of the precursors. Middleton et al. (1985), submitted in 1984, was another independent development motivated by robotics, but using repetitive control (RC). The origins of repetitive control had different motivation, and early works include Inoue et al. (1981), Omata et al. (1984), Hara et al. (1985 a, b), Nakano and Hara (1986) and Tomizuka et al. (1989). These two fields appear to be very different in the literature, but it is suggested here that for practical use they are not really different. Neither field has seen wide application in engineering practice. This paper presents one person’s view of what has prevented wide engineering applications, and gives methods of how to address the issues. It summarizes in a unified manner, developments in many papers by the author and co-workers (Longman 1998). As with the original works in 1984, other authors make use of similar ideas, but we will not attempt here to track the various origins. The purpose of the paper is to develop from first principles what is required of ILC/RC designs in order to be widely useful in engineering practice, and then present ILC/RC laws that meet the requirements.

The iterative learning control problem considers that the control task is to perform a specific tracking command many times. Between each command application, the system is returned to the same initial condition which is on the desired trajectory. The formulation used here considers that there is a feedback controller, and the learning law simply adjusts the command to the feedback controller from one iteration to the next, in order to decrease tracking error. Feedback control tracking error comes from several sources. First, they make deterministic, repeatable errors in following general tracking commands. This is characterized mathematically by the following: the command determines the forcing function to the control system differential (or difference) equation, and the response is a convolution integral (or sum) of the command. As one would expect, this integral (sum) of the command is almost never equal to the command. Second, there are often deterministic disturbances that occur each time the same command is given, e.g. the history of torques from gravity on a robot link as it follows a specific trajectory through the workspace. Third, there will always be some random disturbance errors. It is the aim of the ILC methods in this paper to eliminate as much of the deterministic errors as possible. We note that there are ILC formulations that instead of aiming for zero deterministic error, aim for minimum quadratic cost (Longman et al. 1989, Longman and Chang 1990, Frueh and Phan 2000). Another basic issue is the question of what does one presume to know about the system when designing the ILC. This paper aims to accomplish the design purely from an experimental frequency response plot. This represents one end of the spectrum of interplay between system identification and ILC design. Other approaches are discussed in Elci et al. (1994c), Longman and Wang (1996) and Phan and Frueh (1998). Real time identification or adaptation is treated in Lee et al. (1994), Wen and Longman (1997) and Phan and Longman (1988 b, 1989).

In repetitive control, the command to be executed is a periodic function of time. Again, there may be deterministic disturbances that have the same period. For
example, the period would be the same for gravity torque disturbance on a robot link performing a periodic motion in the work space. There is no returning of the system to the same initial condition before the start of the next period, and thus transients can propagate across periods. Also, changes in control actions made near the end of one period influence the error at the start of the next. This makes the true stability boundary of ILC and RC very different. A special case of repetitive control, probably corresponding to the largest class of applications, has a constant command, but there is a periodic disturbance. A constant command happens to be periodic with any period, and is therefore periodic with the disturbance period. It is the task of the repetitive controller to eliminate the effects of the periodic disturbances on the control system output.

2. Maximizing the impact of ILC/RC in engineering practice

The approach used to maximize usefulness in practice can be characterized as follows.

2.1. Use a linear ILC/RC formulation

The vast majority of practical control problems are addressed by linear control methods. This is true even when the actual problem is known to be non-linear. Hence, here linear ILC and linear RC are considered. In the same manner that non-linear root finding problems can be effectively solved by the Newton–Raphson iteration which uses linear equations, iterative learning control and repetitive control based on linear thinking may solve non-linear problems as well. In addition, in engineering practice one uses the simplest approach that works. Non-linear formulations in ILC usually result in very complicated control equations, for example making use of the full non-linear equations of robot dynamics. Yet, a simple linear learning law was applied to the robot in figure 1 producing the RMS tracking error as a function of repetitions shown in figure 2. The desired trajectory was a large-angle high-speed manoeuvre. The linear ILC law uses one gain and one cut-off frequency, the same for all axes, applied as if each joint was decoupled from the remaining joints. The error is decreased by a factor of 100 in eight repetitions (Elci et al. 1994 a). Figure 3 shows RMS errors when a simple compensator is added, and the error is decreased by a factor of nearly 1000 in about 12 repetitions (Elci et al. 1994 b). This is close to the reproducibility level of the robot, which is the limit of possible improvement. Therefore, no more complicated ILC law could do significantly better. This demonstrates the effectiveness of linear ILC even on non-linear, highly coupled robot dynamic equations.

2.2. Create discrete time ILC/RC laws

ILC and RC make use of the error measured in the previous repetition or period to adjust the control.
action. Storing this information, retrieving it, and perhaps processing it, requires the use of a digital computer on line. Hence, it is best to start with a discrete time ILC/RC formulation.

2.3. **Use the existing feedback controller**

Much of learning control theory chooses to simultaneously create the feedback control law and the learning control law. Such an approach will highly constrain the number of practical applications. Here we work with methods that can apply to any existing feedback control system, and simply learn to improve its performance. In the case of robots or other purchased hardware, using a law that simultaneously creates both, requires that the manufacturer be the one to implement ILC/RC. Here, it can simply be the user of the robot that applies ILC/RC. There are many more users than manufacturers.

2.4. **Adjust the command to the feedback controller, not the manipulated variable**

Often in ILC and RC the manipulated variable, e.g. the torque applied to a robot link, is adjusted by the learning process. This requires going into the existing controller and modifying the signal it sends to actuators. Here such complexity is avoided. The ILC/RC simply adjusts the command given to the existing feedback control system. One can show that these two approaches are essentially equivalent mathematically (Solcz and Longman 1992) so here the approach that is by far the easiest to apply is chosen.

2.5. **Make simple ILC/RC laws with a small number of parameters to adjust**

The largest classes of feedback control laws in practice are, proportional (P), integral (I), proportional plus derivative (PD) and PID. These controllers are simple, require the control system designer to adjust only one, two, or three parameters, and the manner of doing the adjustment is relatively straightforward. The approach here aims to do something analogous for ILC/RC, presenting laws with only a few parameters to adjust in well defined ways.

2.6. **Make use of typical knowledge about the feedback control system behaviour**

One should not try to make a universal learning controller that (in theory) works on all systems. To get good performance in practice, the ILC/RC design should make use of easily obtainable information about the existing feedback control system dynamics. It is reasonable that the designer first perform a frequency response test to produce Bode plots going up to the Nyquist frequency. These plots are easy to obtain, and any practicing control system designer will feel very comfortable working with them. Then the ILC/RC should be designed to work based as directly as possible on this description of system behaviour. This avoids the usual process of trying to fit the data with a mathematical model (e.g. a higher order difference equation model, or a state space difference equation model). Such models fail to exactly fit data introducing extra discrepancy in performance by comparison to the real world, and they constrain the type of behaviour, for example by choice of the order of the model. By staying as close as possible to the data in the frequency response plot these sources of error are eliminated.

2.7. **Guarantee good learning transients**

It is shown that simply satisfying the true stability condition for ILC is of little value in ensuring a practical learning process. Having direct control over the learning transients in ILC/RC is fundamental. The ILC/RC laws described here, address this by aiming for monotonic decay of all frequency components of the error (up to a chosen cut-off frequency). The designer knows the rate of decay for each frequency, and can make adjustments to influence the rate.

2.8. **Guarantee long term stability**

It is demonstrated below that linear ILC/RC can easily exhibit very long term instabilities, not evident for perhaps thousands of repetitions. The ILC/RC laws give the designer methods to kill such instabilities, by adjusting the appropriate parameter(s).

2.9. **Make an impedance match with the knowledge base of practicing control engineers**

To maximize the use of ILC/RC in engineering practice, making such an impedance match allows the control system designer to grasp the concepts fast and know how to apply them immediately. By connecting to the Bode plot information, we create such a match.

3. **Two test beds for experimental demonstrations**

To demonstrate the effectiveness of ILC/RC laws, experimental results are presented from two experimental test beds. ILC tests are on the robot in figure 1, a Robotics Research Corporation K-Series 8071HP seven degree-of-freedom robot. The same desired trajectory is specified for each link, a cycloidal path increasing the joint angle from zero to 90°, and then the cycloidal path is reversed to come back to zero again (the endpoints are connected by a polynomial having zero first and second derivatives at each end). When all seven joints are executing this curve at the same time, it creates a large motion through the workspace. The timing of the
trajectory is made such that the base joints reach the maximum velocity allowed by the manufacturer, 55° per second, so that non-linear interactions between joints such as centrifugal and Coriolis effects are maximized. The maximum tracking error when the commercial feedback controllers are commanded this trajectory simultaneously is about 9° for all joints. Robot manufacturers advertise the high repeatability of their hardware, and they also quote much looser numbers for positioning accuracy, by which they mean positioning under static conditions. They normally do not address the issue of accuracy during dynamic motion, partly because there is no way to characterize the large range of dynamic situations, and partly because it is very typical to experience large errors such as the 9° in this case. Hence, the need for ILC. The learning control laws tested are implemented on the feedback controllers on each robot link individually, i.e. in a decentralized manner, so that centrifugal torques on a link produced by the motion of another link are handled as disturbances. The learning control laws are implemented at the 400 Hz sample rate of the feedback controllers, bypassing the upper level path generator. More complete information on these experiments can be found in Elci et al. (1994 a,b,c) and Lee-Glauser et al. (1996).

The RC experiments are on a constant velocity double reduction timing belt drive system, figure 4 (Hsin et al. 1997 a,b). Timing belt drives are easy ways to produce a gear ratio to allow a dc motor to run at higher velocities and produce more torque. A dc motor drives an input shaft with a gear on it. A timing belt (a belt having teeth) goes between this gear and a larger gear on one end of an idler shaft. A second timing belt connects a smaller gear on the other end of this shaft to a larger gear on the output shaft. The overall gear ratio from dc motor to output shaft is 8 to 1. It is desired to have a very accurate constant velocity of the output shaft. A well designed feedback controller is in place. The error spectrum in its output shaft velocity is shown in figure 5, and the RMS error in output velocity is between 6 and 7 × 10⁻⁴ m/s. The physical causes of velocity error include: inaccuracies in the machining or mounting of shafts which produce errors with a period of one rotation of the shaft and harmonics, similar errors that are periodic with the period of each timing belt rotation, and tooth meshing dynamics of the belt teeth with the gear teeth. The shaft and belt speeds are related by gear ratios so there is a common error period. The objective of the repetitive controller is to eliminate these velocity errors.

4. Linear iterative learning control

4.1. Statement of the linear learning control problem

This section starts with a general modern control formulation for linear learning control and develops the associated stability condition for convergence to zero tracking error (Phan and Longman 1988 a). Let

\[ x(k+1) = Ax(k) + Bu(k) + w(k); \]
\[ y(k) = Cx(k); \quad k = 0, 1, 2, 3, \ldots, p-1 \]

represent the closed loop dynamics of the feedback control system, with \( u(k) \) being the command input, and \( y(k) \) being the output. For simplicity we consider single-input, single-output (SISO) problems. One supposes that \( CB \neq 0 \) which is true for nearly all systems as discussed in §5. The \( w(k) \) represents any deterministic disturbance that appears every repetition. The desired trajectory \( y^*(k) \) is \( p \) steps long. Define the \( p \) step histories of the input and output according to
\[ u_j = [u_j(0) \ u_j(1) \ \cdots \ u_j(p-1)]^T \]  
\[ y_j = [y_j(1) \ y_j(2) \ \cdots \ y_j(p)]^T \]

Underbars are used analogously on other variables. The subscript \( j \) indicates the \( j \)th repetition. At repetition 0 one normally applies the desired command as the input \( u_0 \). Then, a rather general linear learning control law makes a change in the command each repetition that is a linear combination of the error \( e_{j-1} \) measured in the previous repetition, where \( e_j = y^*_j - y_j \), i.e.

\[ u_j = u_{j-1} + L e_{j-1} \]  

where \( L \) is a matrix of learning gains. The control laws developed below have this form, but then add to it some form of filtering. Phan et al. (2000) discuss more general forms of ILC laws, and shows that very often there is an equivalent law of the form (4).

### 4.2. Error propagation from repetition to repetition

To determine how the error evolves with repetitions, define a backward difference operator \( \delta_j \) in the repetition variable \( j \). For any variable \( z(k) \) at time step \( k \), \( \delta_j z(k) = z_j(k) - z_{j-1}(k) \). Apply this operator to the convolution sum solution of difference equation (1),

\[ y(k) = CA^k x(0) + \sum_{i=0}^{k-1} CA^{k-i-1} Bu(i) + \sum_{i=0}^{k-1} CA^{k-i-1} w(i) \]

and observe that \( \delta_j x(0) = 0 \), \( \delta_j w(i) = 0 \), because initial conditions and the disturbances are assumed the same every repetition. Package the results in matrix form for all time steps of a repetition to obtain

\[ \delta_j y = P \delta_j u \]  

where

\[ P = \begin{bmatrix} CB & 0 & 0 & \cdots & 0 \\ CAB & CB & 0 & \cdots & 0 \\ CA^2 B & CAB & CB & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ CA^{-1} B & CA^{p-2} B & CA^{p-3} B & \cdots & CB \end{bmatrix} \]

Equations (6) and (7) show how the output changes when one changes the input. Notice that \( \delta_j y = -\delta_j e_j \), and combine (6) and (4) to obtain the evolution of the error history

\[ e_j = (I - PL) e_{j-1} = (I - PL)^j e_0 \]

where \( I \) is the identity matrix (Phan and Longman 1988a).

### 4.3. Different forms of the learning gain matrix

The simplest form of linear iterative learning control can be termed integral control based learning. It uses a discrete version of an integral, running in repetitions, one ‘integral’ for each time step, and adds this to the command. As in integral control, it will not allow the error at that time step to remain constant, building up the corrective action until the error goes away (or the process goes unstable). The control law is

\[ u_{j+1}(k) = u_j(k) + \phi e_j(k+1) = u_0(k) + \phi \sum_{i=0}^{j} e_i(k+1) \]

The centre of this equation gives the law as it would be implemented in recursive form, and the right part gives the summed form showing the ‘integral’ action. Those with experience in classical control may find the integral control analogy appealing and intuitive. There is a second simple and intuitive interpretation. If the robot link was too degrees too low at time step \( k \) during the last repetition, add two degrees (or a gain \( \phi \) times two degrees) to the command for the next repetition at the appropriate time step (i.e. at step \( k-1 \)). Comparing to equation (4) the learning gain matrix \( L \) is diagonal with \( \phi \) along the diagonal. There are many other ways to approach the choice of the gains in \( L \). Some of the possible choices produce the following forms for the matrix.

(i) The pure integral control based ILC produces, \( L = \text{diag}(\phi, \phi, \ldots) \) (Phan and Longman 1988a).

(ii) The contraction mapping learning control law makes \( L \) upper triangular (Jang and Longman 1994, 1996a) and produces monotonic decay of the Euclidian norm of the tracking error with repetitions.

(iii) Learning laws using a causal compensator have lower triangular \( L \), e.g. Elci et al. (1994b). The compensator is used to produce good learning transients. Also, ILC laws that use a system inverse have this form, e.g. Lee-Glauser et al. (1996), but one must be careful with this. Inverting a scalar difference equation model usually involves solving an unstable difference equation, because of zeros outside the unit circle. The corresponding inversion of the \( P \) matrix above reflects this difficulty in ill-conditioning.

(iv) The phase cancellation learning control law produces a full matrix of learning gains (Elci et al. 1994c). Also, the learning control law based on a partial isometry is of this form Jang and Longman (1996b). The phase cancellation law produces monotonic decay of the amplitudes of the errors for each frequency, based on steady
state frequency behaviour. The partial isometry law produced the same monotonic decay on trajectories that are so short that steady state frequency response thinking is not appropriate.

(v) The linear phase lead learning control law is like integral control based learning, but the non-zero diagonal is shifted toward the upper right of the matrix (Wang and Longman 1996). The linear phase lead acts as a simple compensator to produce monotonic decay up to higher frequencies.

(vi) The linear phase lead with triangular windowing has this same shift, but instead of having only one non-zero diagonal, it has a non-zero band (Wang and Longman 1996, Hsin et al. 1997b). The triangular window is used to produce a frequency cut off with no phase shift.

(vii) Linear phase lead with causal Butterworth low pass filtering fills up everything in the matrix below the upper shifted diagonal (Hsin et al. 1997a). The low pass filter again serves as a cutoff, and this time the linear phase lead is adjusted to also compensate for the phase lag of the Butterworth filter.

The ILC/RC laws suggested below for practical application, modify these various forms of ILC by including a frequency based cut-off of the learning. This can be a filter on the error itself, or a filter of the command just before it is applied to the system, or a filter of just the learning control part of the command before it is applied.

4.4. A necessary and sufficient condition for convergence to zero tracking error in ILC

From (8), a chosen learning gain matrix $L$ will cause all possible initial tracking errors to converge to zero, if and only if all eigenvalues of the matrix $(I - PL)$ are less than one in magnitude

$$|\lambda_i(I - PL)| < 1 \quad \forall i \quad (10)$$

In order to be complete, it is shown here that this defines the true stability boundary for linear ILC laws (4). All square matrices are either diagonalizable by a similarity transformation, or they can be put in Jordan canonical form. Suppose first that $M$ diagonalizes $(I - PL) = M^{-1}AM$, and define $g_i^* = Mg_i$. Since $M$ is invertible, $g_i^*$ is zero if and only if $g_i^*$ is zero. Writing (8) in terms of this transformed error gives $g_i^* = A^ig_0^*$, from which it is clear that all eigenvalues on the diagonal of $A$ must be less than one in magnitude for convergence to zero from all initial errors. When a Jordan canonical form applies, $(I - PL) = M^{-1}JM$. The $J^j$ in $g_i^* = J^jg_0^*$ is block diagonal, and raising a square block diagonal matrix to the $j$th power is the same as the matrix with its diagonal blocks raised to the $j$th power. A generic such block has the form $(\lambda I + J)^j$ where $J$ is all zero except for a diagonal of ones immediately above the true diagonal. Note that if $J$ is an $r$ by $r$ matrix, then $J^r$ is zero, and

$$ (\lambda I + J)^j = \lambda^j I + a_1\lambda^{j-1} J + \ldots + a_r\lambda^{j-r+1} J^{r-1} \quad (11) $$

for appropriate combinatorial values for the coefficients $a_1$ and $a_r$. Again, as $j$ goes to infinity, we need that the eigenvalue $\lambda$ to be less than one in magnitude, in order for all possible initial errors to converge to zero as the repetitions progress.

5. The promise and failure of a universal learning controller

All lower triangular forms of learning control law $L$, have a very powerful, robust stability result. Suppose the components on the diagonal of $L$ are given by $\phi$. Then $(I - PL)$ is lower triangular with the entries on the diagonal equal to $1 - CB\phi$. The diagonal elements of a lower triangular matrix are the eigenvalues. Therefore, (10) says the ILC controller will be stable, and converge to zero tracking error for all initial errors, if and only if the learning gain $\phi$ satisfies

$$ 0 < CB\phi < 2 \quad (12) $$

The following comments suggest that this is a very easy condition to satisfy.

(i) Typical systems have $CB \neq 0$. When (1) comes from discretizing a continuous time system, it will not normally be zero. The value $CB$ is the response at time step one for a unit pulse input at step zero. During this time step, the input–output relationship of the continuous time system is the unit step response, which for typical asymptotically stable systems will not cross zero no matter how long you wait before the next time step starts. When the control system is digital, in most cases one can make the same conclusion. The possible exception to $CB \neq 0$ occurs when there are extra delays in the system, for example from having more than one digital part in the loop. In this case one can determine what the true time delay is in the system, and then reformulate the problem so that the desired trajectory is specified starting from the first time step for which an input at zero can influence the output. Then $CAB$ for the appropriate $i$ takes the place of $CB$ in (12), and again we have the same strong result.

(ii) Normally you know the sign of $CB$, since it is the unit step response after one sample time. For a typical feedback control system, when a positive step input is commanded, the response
starts in the positive direction, meaning that $CB$ is positive. The possible exception is non-minimum phase systems (with the zeros outside the unit circle being from the control or dynamics, and not those introduced by discretization). Normally one knows when the system is non-minimum phase.

(iii) As the sample time goes to zero, the value of the discrete time $B$ will go to zero according to

$$CB = C \left[ \int_0^T e^{A_c \tau} B_c \, d\tau \right] = C [B_c T + \frac{1}{2} A_c B_c T^2 + \cdots]$$

where $A_c$ and $B_c$ are the continuous time system matrices. Therefore the range of values of $\phi$ satisfying (12) increases as the sample time decreases.

One can now state the following conclusion. Provided $CB \neq 0$, and you know the sign, then for all learning gains $\phi$ (with sign equal to that of $CB$) sufficiently close to zero, any lower triangular learning law $L$ with $\phi$ on the diagonal, will produce zero tracking error in the limit as the repetitions go to infinity. Furthermore, as the step size gets small, the range of values of $\phi$ in (12) that produce convergence tends toward infinity.

This result is independent of the system dynamics appearing in matrix $A$. Thus the learning law is guaranteed to produce zero tracking error when applied to nearly all feedback control systems. It is like an ideal black box learning controller, or a universal learning controller, just connect it up to nearly any system, turn it on, and after a while you get zero tracking error. The result can be made even stronger by use of the alternating sign learning control law (Chang et al. 1992, Longman and Huang 1996), that eliminates the need to know the sign of $CB$. This law uses $+\phi, -\phi, +\phi - \phi, \ldots$ as the learning gains on the diagonal of $L$ during the first repetition, and then $-\phi, +\phi, -\phi, +\phi, \ldots$ the second repetition, etc. The sign of $CB$ and the sign of $\phi$ become irrelevant because the diagonal elements after two repetitions are $(1 - CB\phi)(1 + CB\phi) = 1 - (CB)^2 \phi^2$ which only depends on the squares of these two quantities.

And both of these results also apply to nearly all non-linear systems as well with digital input through a hold device (Longman et al. 1992). The main requirement is the satisfaction of a Lipschitz condition, something that one expects for differential equations governing feedback control systems.

One should be suspicious of universal controllers—if they work in practice, there would be no need for control engineers. The following gives some indication of how the integral control based learning of item (i) §4.3 with guaranteed convergence to zero tracking error, fails to be practical. When it was applied in experiments to the Robotic Research Corporation robot with $\phi = 1$, the error decreased for about 10 repetitions, and then started to increase. By repetition 15, the hardware was making so much noise that one did not dare go further, for fear of damaging the robot. Computer simulations were made to simulate what would have happened had the repetitions been continued—to see how many repetitions would have been needed to reach zero tracking error. The simulation used a third order linear model of the input-output relationship of one robot joint: the dc gain is unity, there is a real pole with a break frequency at 1.4 Hz, and a complex conjugate pair with undamped natural frequency of 5.9 Hz and damping ratio 0.5. The simulation using the 400 Hz sample rate produced computer exponent overflow. The problem was shortened to a 1 s trajectory, and a 100 Hz sample rate was used, with a desired trajectory $y_d(t) = \sin (2\pi t)$ (Longman and Songchon 1999). The result is shown in figure 6. The RMS of the tracking error decreases from 0.4330 in repetition 0, to a minimum at repetition 7 of 0.1402. The mathematics above proves that the error will converge to zero, but after repetition 7 the error started to grow, reaching a maximum RMS error level of $1.1991 \times 10^{51}$ at repetition 62132. Then the error decreases and ultimately reaches a numerical zero. For example, at repetition $3 \times 10^5$ the error has reached $1.3145 \times 10^{-48}$. If you have a computer that can handle large enough exponents on the way to zero error, you can see that the mathematical proof is correct.

One concludes that, unlike some other fields where guaranteeing convergence/stability is a significant accomplishment, in linear ILC it is nearly meaningless when it comes to practical applications. It is easy to
guarantee convergence, but what is hard is to get reasonable transients during the learning process.

6. Conditions for good learning transients in ILC

6.1. Approximate condition for monotonic decay of tracking error

If one can guarantee monotonic convergence, then one can guarantee good transients. This section and the next section follow Elci et al. (1994 b). Take the transform of (1) for the fth repetition

\[ Y_j(z) = G(z)U_j(z) + C(zI - A)^{-1}z\alpha(0) + C(zI - A)^{-1}W(z) \]

(14)

where \( G(z) = C(zI - A)^{-1}B \). The error is \( E_j(z) = Y^*(z) - Y_j(z) \). This and the next section consider the situation where the learning matrix \( L \) is lower triangular with the same elements for all entries of a diagonal, for all diagonals, so that the \( L \) is generated by a causal time invariant difference equation with transfer function \( \Phi(z) \). The entries in matrix \( L \) are the Markov parameters from the unit pulse response history of the difference equation. This equation operates on the error, and in the ILC problem the initial value of the error is zero. Then the transform of equation (4) is

\[ U_j(z) = U_{j-1}(z) + z\Phi(z)E_{j-1}(z), \]

including the initial conditions. Combining equations gives

\[
\begin{align*}
E_j(z) &= [1 - z\Phi(z)G(z)]E_{j-1}(z) \\
Y_j(z) &= [1 - z\Phi(z)G(z)]Y_{j-1}(z) + z\Phi(z)G(z)Y^*(z)
\end{align*}
\]

(15)

From the first of these, \([1 - z\Phi(z)G(z)]\) represents a transfer function from the error at repetition \( j - 1 \) to the error at repetition \( j \). If we substitute \( z = \exp(i\omega T) \) we get the frequency transfer function which for each frequency is a complex number that can be written as \( M(\omega) = \exp(i\theta(\omega)) \). Suppose one decomposes \( E_{j-1}(z) \) into its discrete frequency components. The corresponding components in \( E_j(z) \) have their amplitudes multiplied by the associated \( M(\omega) \), and their phases shifted by \( \theta(\omega) \). By imposing the condition that \( M(\omega) < 1 \) or

\[ |1 - \exp(i\omega T)\Phi(\exp(i\omega T))G(\exp(i\omega T))| < 1 \]

(16)

for all \( \omega \) up to Nyquist frequency, one assures the amplitudes of all frequency components decay monotonically every repetition, provided we are looking at steady state response. This suggests convergence to zero tracking error, but the reasoning is not rigorous because the conclusions only apply to parts of the trajectory for which steady state frequency response describes the input–output relation. In learning control one is dealing with finite time problems, and hence, technically one never reaches steady state response.

6.2. Relationship to the true stability boundary

In figure 7 one divides the time interval from time step 0 to time step \( p \) into regions. Region 1 corresponds to one settling time of the feedback control system, usually taken as four time constants for the longest time constant in the transient response. Once one passes this amount of time, the effects of the initial conditions have become small, so that the remaining time as far as the feedback control system is concerned, Region 2, can be modelled based on steady state frequency response thinking. Therefore, when one satisfies (16) one aims to make the tracking error in Region 2 decay monotonically as the repetitions progress. Note that condition (16) is very much dependent on the system dynamics in matrix \( A \).

Now consider the mechanism for convergence that determines the true stability boundary (12) for these lower triangular learning matrices \( L \) (Huang and Longman 1996). This mechanism must be unrelated to the dynamics of the system in matrix \( A \). Write the first few time steps of equation (8), and for simplicity consider a diagonal \( L \)

\[
\begin{align*}
e_j(1) &= (1 - CB\phi)e_{j-1}(1) \\
e_j(2) &= (1 - CB\phi)e_{j-1}(2) - CAB\phi e_{j-1}(1) \\
e_j(3) &= (1 - CB\phi)e_{j-1}(3) - CAB\phi e_{j-1}(2) - CA^2B\phi e_{j-1}(1)
\end{align*}
\]

(17)

For the first time step, one keeps multiplying by \((1 - CB\phi)\) every repetition, and if this number is less than one in magnitude (equation (12)) then the magnitude of this error decays monotonically with repetitions. Once \( e_{j-1}(1) \) has become essentially zero, then the second equation in (17) drops its last term, and from then on \( e_{j-1}(2) \) will decay monotonically, etc. Thus, the convergence progresses in a wave, starting from the first time step and progressing step by step through the \( p \)
step process. Represent the location of the ‘wave front’, the time step for which the errors at all previous time steps have become essentially zero, using Region 3 in figure 7. Thus Region 3 defines a transient region for the learning process that grows, and once Region 3 is larger than Region 1, the steady state frequency response modelling apples only in Region 4. What happens in Region 4 is irrelevant to convergence of the learning process, and hence condition (16) which only applies in Region 4 is irrelevant to convergence. Region 3 will eventually grow to include everything.

In learning control the difference between the frequency response condition for convergence (16) and the true stability boundary given by (12), is very large, with one depending on the system dynamics in A and the other not. In fact, it is not obvious that satisfying (16) necessarily implies convergence. For example, perhaps the settling time is longer than $p$ time steps, so that no part of the trajectory can be modelled by steady state frequency response. To determine the status of the frequency response condition (16) as a stability condition (Elci et al. 1994 b), sum the geometric series in (15) to obtain

$$E_j(z) = \left[1 - z\Phi(z)G(z)\right]/E_0(z)$$
$$Y_j(z) = \left[1 - z\Phi(z)G(z)\right]/Y_0(z)$$

where $\phi(0)$ is defined as the entry on the diagonal of $L$, and the other $\phi$ are on the subdiagonals successively. Substitute this into (19), and use a very small $\varepsilon$ so that one can neglect terms that depend on $z^{-1}$ compared to terms independent of $z$, and then (19) says that we need to satisfy $|1 - CB\phi(0)| < 1$, which is the same as (12) (or (10) specialized to lower triangular $L$). We conclude that equations (10), (12) and (19) define the true stability boundary, and the frequency response condition (16) is a sufficient condition for stability. We conclude: by the development of (16), satisfying (16) means that there will be monotonic decay in Region 4. In addition, the above discussion says that satisfying (16) indicates that Region 3 grows to fill the whole time interval, although the error history in this region need not be monotonic. Thus, there are two mechanisms for convergence operating simultaneously, when (16) is satisfied.

6.3. Relationship to the exact Euclidean norm
monotonic decay condition

The frequency response condition (16) indicates monotonic decay in Region 4, but not for the entire trajectory. In this section we consider what is required to guarantee monotonic decay for the entire trajectory, in terms of some error norm. One considers the Euclidean norm as the most natural (Jang and Longman 1994, 1996 a). One can of course examine the maximum component norm as in Oh et al. (1997), or other norms. Define the matrix $H = I - PL$, and suppose that it has a singular value decomposition $H = U S V^T$ where $U$ and $V$ are unitary matrices and $S$ is the diagonal matrix of non-negative singular values $\sigma_i$. Then equation (8) can be written as $U^T \xi_i = SV^T \xi_{i-1}$. Multiplication of an error by a unitary matrix does not change the Euclidean norm of the error, so that the largest the norm of $\xi_i$ can be is the norm of $\xi_{i-1}$ multiplied by the largest singular value, i.e.

$$\|\xi_i\| \leq \max_i (\sigma_i)\|\xi_{i-1}\|$$

(21)

Various iterative learning control laws have been developed to satisfy this condition. This includes the contraction mapping learning control law of Jang et al. (1994, 1996 a) and Lee-Glauser et al. (1996) that uses the learning matrix $L = sP^T$ with the scalar $s$ sufficiently small (note that the comments on robustness of this approach are not valid). It also includes the learning control law based on a partial isometry (Jang and Longman 1996 b), with $L = sV_P U_P^T$, where $V_P$ and $U_P$ are from the singular value decomposition of $P$.

Now consider the relationship between the true monotonic decay condition (21) and the approximate monotonic decay condition (16) developed based on steady state frequency response (Jang and Longman
Again consider matrices $L$ that are lower triangular with all entries in each individual diagonal being identical, so that they correspond to causal time invariant filters. Then $H = I - PL$ is lower triangular and has the same identical entry property. Analogous to the first of (20), there is associated with this matrix, a Laurent series $H(z) = h_0 + h_1 z^{-1} + h_2 z^{-2} + \cdots$, where we introduce the $z$ argument to distinguish the series from the matrix. The transpose of matrix $H$ is upper triangular, and has associated with it a Laurent series $H(z^{-1}) = h_0 + h_1 z + h_2 z^2 + \cdots$. The singular values of $H$ are the square roots of the eigenvalues of $HH^T$. This matrix does not have the identical entry property along each diagonal. However, since the $h_i$ are Markov parameters of an asymptotically stable system (when $G \ldots \hat{z}^i$ and $F \ldots \hat{z}^j$ are asymptotically stable transfer functions) they become negligible eventually. Supposing that the matrix is very large compared to the number of time steps needed for the Markov parameters to become negligible (e.g. one settling time), then the identical entry property on diagonals is essentially met, and $HH^T$ becomes a Toeplitz matrix. Toeplitz and Szegő have proved properties of Toeplitz matrices (Grenander and Szegő 1958) that the eigenvalues when the matrix is infinite in size coincide with the set of complex values that the associated Laurent series assumes at evenly spaced points around the unit circle $||z|| = 1$. The Laurent series is $H(z)H(z^{-1})$, and for $z$ on the unit circle $z^{-1}$ corresponds to the complex conjugate. Hence the singular values, as the matrix gets large, converge to the square root of the magnitude of $H(z)H(z^{-1})$ at the associated points on the unit circle. In other words, they converge to the magnitude $M(\omega)$ of the frequency response of $1-\hat{z}^j\Phi(z)G(z)$ at the evenly spaced discrete frequencies. The conclusion is that as the number of time steps $p$ gets large, the condition for monotonic decay of the Euclidean norm of the error (21) converges to the condition (16). Stated in a different way, equation (21) is the finite time version of (16), where (16) only guarantees monotonic decay in Region 4, while (21) guarantees it throughout the $p$ time steps.

6.4. Understanding the source of bad transients in ILC

It is a very common phenomenon to see a learning control law produce substantial decreases in the error in a few repetitions, and then appear to diverge, e.g. figure 6. Huang and Longman (1996) develop intuitive understanding of this phenomenon in a number of ways. Here two approaches are summarized, starting with one in the time domain. Consider the error equation for time step $p$ in equation (17) using integral control based learning

$$e_j(p) = (I - CB\Phi)e_{j-1}(p) - CAB\Phi e_{j-1}(p - 1) - CA^{p-1}B\Phi e_{j-1}(1)$$

Unlike the error at step 1 in equation (17), this equation has $p$ terms summed together. The number of terms that are significant is limited by how long it takes for the unit pulse response to become negligible, i.e. what power $i$ is needed after $A'$ is negligible. With a fast sample time, this can easily be a very large number of terms. Increasing the sample rate can make the behaviour worse. With a settling time of one quarter of a second, and a sample time of 1000 Hz, there would be 250 terms. So one can easily imagine the sum of all of these terms producing a very large error at the end of the trajectory, while waiting for the wave of convergence to arrive. In Region 3 the error decreases, but in Region 4 the phenomenon of (22) can make large errors. What happens in Region 4 has no influence on whether the system ultimately converges.

Now study the convergence from a frequency response point of view, looking at Region 4. To satisfy (16), one needs to have the Nyquist plot of $z^j\Phi(z)G(z)$ for $z$ going around the unit circle, stay inside a circle of radius one centered at $+1$. (Note that this time we write $z^j$. This explicitly indicates the linear phase lead $\gamma - 1$, and it is now explicit rather than buried in the definition of $\Phi$.) The radial distance from the centre of this circle at $+1$ to a point on the Nyquist plot for frequency $\omega$ is $M(\omega)$, and this is the factor by which the amplitude of the error component at this frequency is multiplied each repetition. Hence, violating (16) for some frequencies will create monotonic growth of error in Region 4 for these components, and this error only goes away by having Region 3 grow to include all $p$ time steps. Figure 8 considers the third order model of the feedback controllers for a robot link described earlier. The curves show the amplitude growth or decay with repetitions for

![Figure 8. The decay and growth of frequency components starting from unit amplitude.](image-url)
error components at frequencies 7.76, 2.54, 1.75, 0.99 and 0.322 Hz going from top to bottom among the curves. These curves presume the initial amplitudes are all unity (Huang and Longman 1996). The first two grow and the remaining ones decay. Now adjust these initial amplitudes to representative values for a feedback control system which will usually be rather small at very low frequencies where the control system is effective, is larger at intermediate frequencies, and dies again at higher frequencies where there are no disturbances. Starting the amplitudes of the error frequency components with the actual distribution for a specific problem (using the cycloidal command trajectory as input to the feedback controller), the initial amplitudes for the errors that grow are very small, while the initial amplitudes for errors that decay are large. Summing these curves using these initial amplitudes produces the predicted error versus repetition for Region 4, shown in figure 9. This is quite close to the observed experimental behaviour. Huang and Longman (1996) give two other ways to understand the initial decay followed by growth phenomenon, using root locus thinking and understanding of the size of the coefficients in the solution of the homogeneous difference equation for the error.

6.5. The use of the frequency response condition as a condition for good learning transients

Since it is so much harder to satisfy the sufficient stability condition (16) than it is (10) or (12), one may ask, do I really need to satisfy (16) in my ILC design? In engineering practice one not only needs convergence, but one needs reasonable transients. Condition (16) indicates monotonic decay of each frequency component of the error, when the settling time of the feedback controller is not long compared to the $p$ time steps of the desired trajectory. Monotonic decay is good behaviour. But could I still get good transients by satisfying some more relaxed condition? Usually the answer is no. Consider having a frequency for which the phase angle in the Nyquist plot has reached $-180^\circ$. It is very easy to have this situation. It is equivalent to changing the sign on the learning process for this frequency, producing positive feedback instead of negative feedback. This produces unstable behaviour in Region 4. This can only be tolerated with reasonable transients if the wave of the learning transient arrives quickly. But this wave is usually one time step at a time, and can easily take a long time to arrive. Meanwhile the behaviour in Region 4 keeps growing. This thinking suggests that except for rather specialized situations, one should consider it a necessity to satisfy (16) to get practical learning control laws (Elci et al. 1994 b).

Condition (21) is a more precise condition for monotonic decay, that applies throughout the $p$ step trajectory, not just in Region 4. However, satisfying this condition usually produces time varying coefficient systems, and becomes unmanageable for large $p$. For most applications one settles for trying to satisfy (16). For systems with long settling times compared to $p$ it might be necessary to apply condition (21).

7. Linear repetitive control

7.1. Formulation of the linear repetitive control problem

This and the following sections parallel various learning control sections above, making the corresponding repetitive control version. Let $p$ be the number of time steps in a period of the periodic disturbance and the desired trajectory. The integral control based repetitive control law, analogous to (9), must go back to the appropriate time step of the previous period according to

$$u(k) = u(k-p) + \phi e(k-p+1) \tag{23}$$

where $k$ is the current time step. More general linear repetitive control laws require gains multiplied times errors at additional time steps in the previous period

$$u(k) = u(k-p) + \sum_{i=i_0}^{i} \phi(i)e(k-p+\gamma+i) \tag{24}$$

The $\gamma$ is one in (23), and this accounts for the one time step delay in a typical digital control system between a change in the input and its first influence on the sampled output. The $\gamma$ is set to higher values to produce linear phase lead repetitive controllers. A shift of a chosen number of time steps is a larger percentage of a period as the frequency is increased, making this $\gamma$ produce a linear phase lead in the frequency response. The $i_0$ is usually negative. The summation is a convolution sum, but an alternative is to process the error through a linear difference equation, which might be a compensator or
a causal Butterworth filter. There is of course an equivalence between a difference equation solution and its convolution sum solution, but the second option can be written explicitly as

\[ u(k) = u(k-p) + \hat{e}(k-p+\gamma) \]  \hspace{1cm} (25)

\[ \hat{e}(k-p+\gamma) = \alpha_1\hat{e}(k-p+\gamma-1) + \cdots + \alpha_n\hat{e}(k-p+\gamma-n) - \beta_1e(k-p+\gamma-1) - \cdots - \beta_ne(k-p+\gamma-n) \]  \hspace{1cm} (26)

The z-transform of the discrete time sequence of gains in (24) is

\[ \Phi(z) = \phi(i_0)z^{-i_0} + \cdots + \phi(0)z^0 + \cdots + \phi(i_f)z^{-i_f} \]  \hspace{1cm} (27)

In the case of (25) and (26), the corresponding \( \Phi(z) \) is the z-transfer function of the compensator or filter (26). When needed, we denote \( \Phi(z) \) in terms of its ratio of polynomials, \( \Phi(z) = \frac{\Phi_P(z)}{\Phi_D(z)} \).

7.2. Comparison between ILC and RC formulations

This section examines the relationship between the repetitive control laws discussed here, and the learning gain matrix of (4). The matrix \( L \) analogous to the RC law (24) is

\[
L = \begin{bmatrix}
\phi(1-\gamma) & \cdots & \phi(i_f) & \cdots & 0 \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
\phi(i_0) & \cdots & \phi(1-\gamma) & \cdots & \phi(i_f) \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
0 & \cdots & \phi(i_0) & \cdots & \phi(1-\gamma)
\end{bmatrix}
\]  \hspace{1cm} (28)

The values of \( i_0 \) and \( i_f \) might be so large that the matrix is filled. Or it may start filled, but when the gains get small enough they are truncated. When (25) and (26) is used, the triangle of zeros in the upper right remain, but the rest of the matrix will be filled by the pulse response history of the filter (26), which again might be truncated if needed. Thus, the general repetitive control laws discussed above can correspond to general learning control gain matrices \( L \), with the stipulation that the gains in the matrix within each diagonal are all the same. Without this property, the corresponding RC version would produce time varying repetitive control laws. Most ILC laws have this property, for example, learning laws based on frequency response thinking have the property automatically. One exception to this rule is the learning control law based on a partial isometry (Jang and Longman 1996 b). Hence, most RC laws have a direct ILC counterpart, and vice versa. The main differences are the finite time nature of ILC, and the fact that in learning control one need not compute in real time. One can compute between repetitions, or simply apply the result at the next repetition once it is ready.

Sometimes the time domain multiplication of matrix \( L \) with the error contains many multiplies. An alternative computation for matrices of the form (28), is to perform the product in the \( z \)-transform domain. Since the product represents a convolution sum, one simply takes the transforms of the error and the \( \phi(k) \), multiplies them together and takes the inverse transform (Lee-Glauser et al. 1996). The advantage is that the computation can be much faster. The disadvantage is that it ignores edge effects from the finite size of the matrix.

7.3. Necessary and sufficient conditions for convergence to zero tracking error in RC

7.3.1. The error difference equations. In order to understand the error history for RC laws, we create the \( z \)-transfer functions from the desired trajectory and the repetitive disturbance to the resulting tracking error, and interpret the result in terms of the corresponding difference equation. The output is

\[ Y(z) = G(z)U(z) + V(z), \]

where \( V(z) = C(zI - A)^{-1}W(z) \) is the output from the periodic disturbance. The \( V(z) \) can be considered periodic, since any non-periodic terms are part of the transient response of the presumably stable feedback control system, and thus go to zero. The transform of (24) or (25) and (26) gives

\[ U(z) = \frac{z^\gamma \Phi(z)}{z^p - 1} E(z) \]  \hspace{1cm} (29)

where the error is defined as \( E(z) = Y^*(z) - Y(z) \). Write \( G(z) \) in terms of its numerator and denominator polynomials as \( G(z) = \frac{N(z)}{D(z)} \). Combining these produces

\[ [(z^p - 1)D(z)\Phi_D(z) + z^\gamma N(z)\Phi_P(z)]E(z) \]

\[ = [D(z)\Phi_D(z)(z^p - 1)][Y^*(z) + V(z)] \]  \hspace{1cm} (30)

This can be converted to a difference equation by noting that the terms in square brackets are polynomials, and that a term such as \( z^\delta E(z) \) represents \( e(k+p) \) in the time domain. For periodic commands \( Y^*(z) \) and disturbances \( V(z) \) of period \( p \), the right-hand side of (30) is zero, since \( (z^p - 1)Y^*(z) \) is the command shifted one period ahead, minus the command at the present time. The same is true of the disturbance term once it reaches steady state. Hence the error history is the solution of a homogeneous difference equation.

7.3.2. Conclusions from the time domain viewpoint. The tracking error will converge to zero for all initial conditions (all initial values of the error during period/repetition zero), if and only if all roots of the characteristic polynomial
\begin{equation}
(z^p - 1)D(z)\Phi_D(z) + z^r N(z)\Phi_N(z) = 0 \tag{31}
\end{equation}
are inside the unit circle. Unfortunately, the high order of this polynomial produced by the number of time steps \( p \) in a period, usually makes it impractical to use this condition directly to determine stability, e.g. by actually finding the roots, or by use of the Jury test or the Routh tests with bilinear transformation. It is somewhat easier to use the departure angle condition from root locus plot rules to determine if sufficiently small repetitive control gains produce a process that converges to zero tracking error. Longman and Solcz (1990) start from a base case of a system with a single pole at the origin (on top of the zero of the repetitive controller) This case has a root locus where all roots starting on the unit circle move radially inward. Then the pole at the origin is moved to the location of a true pole of the system, and the remaining poles and zeros are introduced. It is reasonably simple to evaluate the change in departure angle from radially inward that occurs from moving the one pole and adding the remaining poles and zeros. If this change is less than \( \pm 90^\circ \) for all poles on the unit circle (and the feedback control system is stable) then sufficiently small repetitive control gains produce convergence to zero tracking error.

7.3.3. Conclusions from frequency domain stability theory. Now consider the somewhat more practical use of frequency response conditions based on Nyquist stability theory. Divide (23) by \( z^p D(z)\Phi_D(z) \), to obtain \( P(z) \equiv 1 - z^{-p}(1 - z^r \Phi(z)G(z)) = 0 \). The numerator of this expression after being put over a common denominator is the actual characteristic equation of interest, and for asymptotic stability we want no roots outside or on the unit circle. The denominator \( z^p D(z)\Phi_D(z) \) contains the characteristic equation of the closed loop feedback control system, and the characteristic equation of any compensator we choose to use in the repetitive control law. Both should correspond to asymptotically stable systems, so that we know \textit{a priori} there are no roots outside the unit circle. Form a contour in the complex \( z \)-plane that includes all of the plane outside the unit circle, by going around the unit circle in a counterclockwise direction, out to infinity along a branch cut on the negative real axis, then circle at infinity in a clockwise direction, and return along the branch cut. Then for each root of the numerator characteristic outside the unit circle, the angle of \( P(z) \) will decrease by \( 2\pi \) after traversing the contour. If there is no change in the angle of \( P(z) \) then the learning process is convergent.

Now consider instead plotting \( P^*(z) = z^{-p}(1 - z^r \Phi(z)G(z)) \) for \( z \) going around the contour. The number of times this plot encircles the point +1 in a clockwise direction is equal to the number of roots outside the unit circle. The order of the denominator in \( P^*(z) \) will usually be much larger than the order of the numerator because for causal systems \( G(z) \) and \( \Phi(z) \) will generically have one more pole than zero, and then \( p \) is usually a large number. Hence, only \( \gamma \) that are so large that they send one into the next period would violate this condition. With the denominator order larger, the part of the contour at infinity maps into zero. Also, the part of the contour along the branch cut cancels.

Under the above stated conditions, one simply needs to determine the number of times the plot of \( P^*(z) \) encircles +1 for \( z \) going around the unit circle in order to determine stability of the repetitive control system. Simply look at the frequency response of \( P^*(z) \) by substituting \( z = \exp(i\omega T) \) with \( T \) being the sample time, and letting \( \omega \) go up to Nyquist frequency. If this plot together with its complex conjugate reflection, does not encircle the point +1, then the repetitive control system is asymptotically stable, and converges to zero tracking error for all possible initial errors.

7.4. Approximate condition for monotonic decay in RC

This section and the next section are analogous to §§6.1 and 6.2 and follow Huang and Longman (1996). One can rearrange the terms in equation (31) to read
\begin{equation}
z^p E(z) = [1 - z^r \Phi(z)G(z)]E(z) \tag{32}
\end{equation}
One might think of \( E(z) \) as the transform of the error at one repetition, and then \( z^p E(z) \) shifts one period giving the error in the next repetition. By setting \( z = \exp(i\omega T) \) in the square brackets in (32) it becomes the frequency transfer function from one repetition to the next. The equation suggests that requiring
\begin{equation}
|1 - e^{i\omega T}\gamma \Phi(e^{i\omega T})G(e^{i\omega T})| < 1 \tag{33}
\end{equation}
for all \( \omega \) up to Nyquist frequency, would then establish convergence to zero tracking error. This is the same condition for RC as (16) is for ILC (although this time derived with any linear phase lead \( \gamma \) pulled out of the \( \Phi(z) \)).

This thinking is of course quite heuristic (the next section gives a precise conclusion). A frequency transfer function applies only to steady state response, so technically, one cannot use it except after the error has converged. \( E(z) \) is not the transform of the error for one period, but the transform of the error for all time, as is \( z^p E(z) \). Hence, the monotonic decay argument only makes sense in a quasi steady state situation. If one makes significant changes in the repetitive control signal during each period, then it is certainly not steady state. In addition, altering the repetitive control signal toward the end of the previous period changes the initial conditions for the next. Referring to figure 7 as a representation of a period rather than a repetition, it is
clear that one again needs Region 2 to dominate over Region 1 so that the overall behaviour can be viewed in terms of steady state frequency response thinking. Hence, the hidden assumptions are to learn sufficiently slowly and have a system with a settling time short compared to a period. Under such conditions, satisfying (33) should produce monotonic decay of all frequency components of the error.

7.5. Relationship to the true stability boundary in RC

As in the previous ILC sections, an if and only if condition defining the borderline of convergence to zero tracking error has been developed, and a heuristic condition for monotonic decay of the repetitive control transients has been developed. One now asks, does satisfying this heuristically derived monotonic decay condition actually guarantee convergence to zero tracking error, independent of whether the sufficiently slow learning and relative region size conditions described above are satisfied? Above it was determined that (under appropriate assumptions) the repetitive control system is asymptotically stable and converges to zero tracking error for all initial errors, if and only if the plot of $P(z) = z^{-\theta}(1 - z^{-\theta}F(z))$ does not encircle the point +1 for $z$ going around the unit circle. For such $z$, $|z^{-\theta}| = 1$. If one requires that (33) be satisfied, then $|P(z)| < 1$ for all $z$ on the unit circle contour, and one concludes that it would be impossible to encircle the point +1. Therefore, condition (33) is a sufficient condition for asymptotic stability of RC.

In ILC the true stability boundary (12) is unrelated to the system dynamics in matrix $A$, and the monotonic decay condition (16) is heavily dependent on $A$. Hence, there is a large discrepancy between conditions for stability and for monotonic decay. Repetitive control is totally the opposite, the approximate monotonic decay condition is essentially the same as the true stability boundary for typical values of $p$. Violating (33) can very easily send the system unstable. Suppose that (33) is violated for a range of frequencies $\Delta \omega$. Then the phase change in $P(z)$ produced by $z^{-\theta}$ over this interval is $\Delta \omega pT$. The number $p$ is usually a large number, for example in a one second trajectory at a sample rate of 1000 Hz, $p$ is 1000. If $\Delta \omega pT$ corresponds to more that $2\pi$ for some frequency below Nyquist, then $+1$ is encircled, and the repetitive control process is unstable. Hence, satisfying (33) in repetitive control, is not just important for good transients, it is almost a requirement for convergence.

7.6. A comment on the relationship between ILC and RC in engineering practice

Section 6.4 described and explained the commonly observed property of learning control applied to the real world, that initially the error decreases substantially, and then it starts to grow, as if it were unstable. Figure 6 demonstrates that this growth is not likely to be an indicator of mathematical instability, but rather it is just a poor transient while waiting for Region 3 in figure 7 to grow. In repetitive control, the same substantial initial decay of the error is often observed. But this time, when the error starts to grow, it normally is an indicator that the repetitive control system is unstable.

The resetting of the initial condition in ILC creates Region 3 and the stability that it produces. Repetitive control has no such mechanism, and hence, once it starts to diverge, it will continue to do so.

Initially ILC and RC seemed very different, with very different formulations. The stability properties seemed unrelated. Here it is claimed that in practice there is really no difference between linear ILC and RC. To get practical results, one wants to satisfy the same condition, (16) or (33), and hence there is really no distinction in designing and tuning the laws. A repetitive control law for a given system can be used as a learning control law as well. Generally, if one is stable, then the other is stable. On the other hand, if a learning control law has transients that are reasonable in engineering practice, then it most likely satisfies the condition (16) or (33) and hence could serve as a repetitive control law.

Acknowledging this similarity, there are some distinctions between ILC and RC: (1) A seemingly big distinction is that in learning control one can consider the computation to be a batch computation performed between repetitions. If there is not enough time between repetitions, one can always apply the resulting control update at the next repetition after the computation is ready. This produces extra design freedom by comparison to typical repetitive control that operates in real time. However, the distinction disappears when one uses batch process repetitive control, i.e. repetitive control laws that use batch computations applied when ready (Hsin et al. 1997a). (2) The above equivalence is based on satisfying (16) or (33), conditions that apply to time invariant control laws. Learning control is free to make learning gain matrices that vary the entries going down each diagonal. The ILC based on a partial isometry has this form which is needed for monotonic convergence in short trajectories. (3) A third item is that in repetitive control there is an issue of keeping the corrective signal in phase with the period of the error that is to be cancelled. There are three issues. One repetitive control problem with no counterpart in ILC is that the period may not be an integer number of samples times, making it hard to do the updates from the error at the corresponding time in the previous period. When the period is determined by a periodic command, then one is likely to be able to have an integer number of samples per period,
but when it is determined by an external disturbance, this becomes less likely. The second is that the period may not be perfectly known and hence the repetitive control signal has to track the true period to keep in phase. The third is that there are many repetitive control problems where the period is determined by an external source that drifts with time, creating a need for the repetitive control algorithm to track changes of period. Generally, it is best if one can have a signal that indicates to the repetitive controller when the next period starts. In the double reduction timing belt drive, an index pulse was used related to the output shaft angle encoder. In work on eliminating 60 Hz and harmonics on the Jefferson National Accelerator Facility 4 GeV continuous electron beam, a trigger signal from the 60 Hz supply is available (Longman et al. 2000). When no such signal is available, then one can aim to use pattern matching to keep things synchronized as in Longman and Huang (1994). Another approach is to develop disturbance identification methods as demonstrated in Edwards et al. (2000), but this starts to go outside the class of approaches being discussed here. The basis function repetitive control in Longman et al. (2000) is a useful way to deal with the period not being an integer number of sample times.

8. The approach to designing practical linear learning and repetitive controllers

The remainder of this paper makes use of the conclusions above in order to present a road map for designing simple and effective learning and repetitive controllers. The design process works as follows:

(a) Aim to satisfy the approximate monotonic decay condition for the frequency components of the error (33). It is assumed that the system settling time is sufficiently short that this condition is a reasonable indicator of monotonic decay of the frequency components.

(b) It is assumed that there is an existing feedback controller, and that one can conduct frequency response tests on it. The manner of conducting these experiments is described in §9.

(c) A suitable compensator $z^R \phi(z) = \phi z^\gamma$ is chosen. This is accomplished by multiplying the chosen compensator transfer function, converted to a frequency transfer function by substituting $z = e^{i\omega T}$, times the experimental frequency response data $G(e^{i\omega T})$, to produce a plot of the frequency response of $z^R \phi(z)G(z)$. This is done for various choices of compensator parameters, and then those parameters that keep the plot within the unit circle up to the highest frequency, or up to a desired frequency, are chosen. In the case of a pure linear phase lead compensator $z^R \phi(z) = \phi z^\gamma$ where $\phi$ is the learning gain. One adjusts the $\gamma$ and the $\phi$. Larger values of $\phi$ attempt to learn faster, but this effect is limited. Lower values create smaller final error levels in the presence of noise, and will allow a somewhat higher frequency cutoff. For any chosen $\phi$ one adjusts $\gamma$ to keep the plot satisfying (33) to as high a frequency as possible. If one decides to use a causal low pass filter in $\Phi(z)$ to improve the cutoff (see §11.4), then the $\gamma$ is adjusted to compensate for phase lags from this filter as well as $G(z)$ to keep the product inside the unit circle (33) to as high a frequency as possible. Section 10 discusses this process.

(d) In practice it is unreasonable to demand that (33) be satisfied for all frequencies up to Nyquist. Hence, a cutoff is made, to stop the learning process for frequencies that violate condition (33). This means that we do not aim to get zero tracking error, but rather aim for zero tracking error for the frequency components of the error up to the cutoff, aiming to leave frequency components above the cutoff unaltered. Section 11 describes a set of choices of how to make a cutoff, and what to apply the cutoff to.

9. Experiments to perform on the feedback control system

Experiments can easily be performed to produce data for generating the frequency response $G(e^{i\omega T})$ of the feedback controller. Typically one applies a white noise command to the system, takes the discrete Fourier transform of the output, divides it by the discrete Fourier transform of the input, to obtain the frequency transfer function of the system $G(e^{i\omega T})$. This is the result one needs. Normally in obtaining the frequency response one goes one step further and finds the magnitude $M(\omega)$ and phase angle $\theta(\omega)$ and makes the Bode plots. Since performing frequency response tests is a standard procedure, there is considerable expertise available to produce clean results when the data is particularly noisy. One can use long data records to improve the accuracy, and one can turn off the input after some time and allow the output to decay, in order to decrease leakage effects. Often an average of shifted subsets of the data is used to produce clean results. Ideally one should ensure getting reproducible results up to Nyquist frequency, but if this is difficult, the cutoff frequency can be chosen to cutoff below where the results become questionable. For more precise results one can do a sine sweep instead of putting in white noise which produces the most precise results.
Note that when there is a periodic disturbance, the experiment needs to be done correctly. The output is \( Y(z) = G(z)U(z) + V(z) \), with \( V(z) = C(zI - A)^{-1}W(z) \) being the periodic or repeating change in the output due to the plant disturbance \( W(z) \). One can input a signal \( U_1(z) \) such as white noise and record the output \( Y_1(z) \). Then input \( U_2(z) \), the same command multiplied by some factor (making sure in the case of repetitive control that the two commands start at the same time in the periodic disturbance), and record response \( Y_2(z) \). Then difference these two sets of inputs, and the two sets of outputs, to eliminate the repeating disturbances \( V(z) \) from the input–output relationship, and create the frequency response function \( G(e^{j\omega T}) \) from this differenced data.

In order to predict the final error level reached by the controller when there is a cutoff, one needs to compute the transform of the desired trajectory, \( Y^*(e^{j\omega T}) \), and to find the frequency components of the disturbance from \( V(e^{j\omega T}) = Y_2(e^{j\omega T}) - G(e^{j\omega T})U_2(e^{j\omega T}) \). This information is also used to make a good choice of what to filter (§11.3). One would like to know how much improvement is obtained by using the learning/repetitive controller compared to using feedback control alone, so one also applies command \( Y^*(z) \) to the feedback control system, records the error and finds its frequency components.

10. The compensator

10.1. Choice of compensator

Since the Nyquist plot of \( z\phi G(z) \) will essentially always go outside the unit circle centred at +1 for all choices of the learning gain \( \phi \), violating good transient condition (33), it is best to include some form of compensation in the ILC/RC law, picking \( \gamma \) and \( \Phi(z) \) in \( z^\gamma \Phi(z) \) in order to keep the plot inside the unit circle up to higher frequencies. Some of the choices are as follows:

(a) No compensator—The simplest thing to do is to use no compensator and simply cut off the learning when the plot of \( z\phi G(z) \) goes outside the unit circle. This is a very simple control law using only two parameters, the learning gain and the cut-off frequency. Yet this method can be very effective. Without the cutoff, pure integral control based learning produced the bad transients in figure 6 on a mathematical model of one link of the robot. But introducing a cutoff at 3 Hz produces the experimental RMS tracking errors shown in figure 2. In eight learning control iterations the tracking error following the high speed trajectory was reduced by a factor of 100. This amount of improvement is very substantial and can be very significant in engineering practice. No compensator was needed.

(b) Linear phase lead compensator—Introducing a linear phase lead makes a very simple compensator, with one new parameter \( \gamma \) to adjust. In this case, the ILC/RC is \( z^\gamma \Phi(z) = \phi z^\gamma \) where \( \phi \) is the learning gain. To improve the cutoff, one may use \( z^\gamma \Phi(z) \) with \( \Phi(z) \) chosen as a causal low pass filter, and then adjust \( \gamma \) (§11.4). It is suggested here that adding this one extra parameter \( \gamma \) (and possibly this low pass filter) produces the most useful and versatile simple ILC/RC. Only three parameters are needed, the gain, the linear phase lead and the cut-off (and with the filter, the filter order is a parameter). With a cutoff at 160 Hz (produced by a non–zero phase Butterworth low pass filter, quantization stabilized, see §11.5) the error spectrum for the timing belt drive system is shown in figure 10. All peaks of figure 5 below the 240 Hz peak are cancelled producing a very effective repetitive controller. Note that in §11.1 it is suggested that one should not try to cancel the 240 Hz peak to be respectful to the hardware, so this control law cancels all the error that is appropriate to cancel.

(c) Specially tailored compensator—One can design a special compensator for the problem at hand. It is suggested that this be done only when the simpler linear phase lead approach above cannot reach the desired error levels. For most systems there is no need for the extra complexity, and extra design effort involved in a special compensator. One exception is systems with very lightly

Figure 10. Spectrum at repetition 10000, linear phase lead, 160 Hz 18th order non-zero phase Butterworth low pass filter.
damped modes for which the system transfer function phase angle makes a large abrupt drop going across a resonance. Some options for designing specially tailored compensators are as follows:

(i) Sometimes the ILC/RC literature suggests using $\Phi(z) = \tilde{G}^{-1}(z)$ to cancel the poles and zeros of the closed loop transfer function. If the inverse of the transfer function is stable, this can work. The approach is rather aggressive, and in the next items below, less aggressive and hence more robust versions are discussed. There need not be a problem with causality of the control law, because the extra poles from $z^p - 1$ make the compensator causal. Although the literature does not always acknowledge the difficulty, small inaccuracies in the model will usually make this method unstable in practice. With the cutoff methods discussed here, one can stabilize the process in practice, and make it into something that can be practical. The approach relies on a pole zero model, and at high frequencies the model most likely starts deviating substantially from the experimental $G(e^{j\omega})$. Then introducing a cutoff gives robustness to the approach, by adjusting the cutoff to occur before this disagreement sends the plot outside the unit circle. Given good experimental data, one can make this adjustment, and produce a practical ILC/RC system.

(ii) A complication with (i) is that in most cases discrete time models of physical transfer functions have zeros outside the unit circle, making the inverse of the transfer function unstable. In this case, one can invert the transfer function except for the non-minimum phase zeros. To handle these zeros one can cancel their phase as in Tomizuka et al. (1989). Again, a cutoff would normally be needed.

(iii) The approaches in (i) and (ii) attempt to invert the system. At sufficiently high frequencies, the system output becomes very small, and then the inverse is in some manner ill conditioned. Small errors in the magnitude measurement at high frequencies, will get amplified significantly when inverted to create the compensator. One can decrease this sensitivity, adjusting the cancellation to be less aggressive by not fully inverting the magnitude (no longer attempting to learn everything in one repetition), and put the emphasis on cancelling the phase. Someone who enjoys designing compensators, can enjoy making such designs. But the authors suggest going further, and simply use phase cancellation learning control (Elci et al. 1994c, Longman and Wang 1996, Hsin et al. 1997 a) that aims only to have the ILC/RC invert the phase change going through the feedback controller. In ILC or batch process repetitive control, by going to the frequency domain, one can compute this control law reasonably fast. There are some leakage effects. For real time repetitive control, one can compute the learning gains to use in the time domain. Real time applications will limit the number of gains that can be applied each time step, and this can produce side lobes in the resulting window filter. These lobes will normally again require the use of a frequency cutoff, and in the batch process version, one might want a cutoff for robustness. Figure 11 shows the best results obtained using phase cancellation repetitive control implemented in batch update form. In practice a cutoff would be needed but no evidence of this need is yet visible at repetition 50.

10.2. Tuning the compensator

This paper suggests the use of either, no compensator $z^r \Phi(z) = z \phi$, a linear phase lead compensator $z^r \Phi(z) = \phi z^r$, or a linear phase lead compensator $z^r \Phi(z)$ with a causal low pass filter, i.e. $\Phi(z)$ is the product of a learning gain (or repetitive control gain) $\phi$ and a causal low pass filter. This section concentrates on the first two options, and the third is considered in §11.4. In the first, one chooses $\phi$ and then determines the cutoff to use. In the second, one chooses $\phi$, and adjusts $\gamma$ to make the highest possible filter cutoff.

The choice of the gain $\phi$ is not very critical. It can be used to make some minor optimization, but normally it is sufficient to choose a reasonable value and simply use...
it. It is suggested that the maximum reasonable value is the reciprocal of the dc gain of the system. Using this value the ILC/RC will correct all of the error at dc in the next repetition. If the $\phi$ is set to a higher value, then dc components are overcorrected each repetition, converging in an oscillating manner. As the magnitude Bode plot of the feedback controller starts to drop off with increasing frequency, the amount learned each repetition becomes less. The factor by which the error is decreased each repetition for each frequency component is given by the radial distance from the point $+1$ to the Nyquist plot of $z^\prime \phi(z) G(z)$.

The author suggests picking a gain $\phi$ in the range $1/4 \leq \phi \leq 1$, with some preference given to lower values in this range. Since feedback control systems normally have a dc gain of one or slightly less than one, a learning gain of $\phi = 1$ is reasonable, but somewhat on the aggressive side. A lower learning gain will learn less with each repetition, but speed of learning is not usually a critical issue, waiting three or four more repetitions to reach the steady state error level is not usually serious. On the other hand, a smaller learning gain makes the ILC/RC corrective actions less sensitive to noise, and hence can produce a lower final error level (Phan and Longman 1988 a). The lower gain allows some smoothing across previous repetitions. In addition, a lower gain will usually allow a somewhat higher cutoff, so that more of the error is eliminated by the ILC/RC. On the other hand, if the disturbance environment can change, a higher learning gain can give a faster reaction time to such a change. There is usually a gain in the above range for which the learning is the fastest, and gains above that value, although they aim to learn more with each repetition, in fact the learning takes longer to converge.

When no compensator is used, the only thing remaining to do after picking the gain $\phi$ is to determine the cutoff. This is done by plotting $z^\prime \phi G(z) = \phi e^{i\omega T} G(e^{i\omega T})$ using the experimental $G(e^{i\omega T})$, and observing the frequency at which the plot goes outside the unit circle. The cutoff is then picked conservatively at a value somewhat below this frequency, because the cutoff will not be perfect. Figure 12 shows the Nyquist plot produced in this manner from data ($\phi = 1$), for the timing belt drive system. Without a compensator, the amount of error that can be eliminated is the frequency components up to the cutoff chosen. In some situations, this captures the majority of the error (for the robot experiments it represented a decrease in error by a factor of 100), and hence this is a viable design.

But introducing a linear phase lead is very simple and can improve the performance significantly. Figure 13 shows the plot of $z^\prime \phi G(z) = 0.5(e^{i\omega T})^\gamma G(e^{i\omega T})$ from experimental data, using a learning gain of 1/2 and a linear phase lead set to $\gamma = 6$ time steps (Hsin et al. 1997 a). This also goes outside the unit circle, but at a much higher frequency so that much more of the error can be eliminated. The amount it goes outside is imperceptible on the plot. To determine the cutoff, one must plot the data in a different manner, plotting the radial distance from the centre of the circle at $+1$ to the Nyquist plot for each frequency. When this distance becomes greater than one, the plot has gone outside the unit circle. Figure 14 gives an example of such a plot, in this case it corresponds to the third order model for one link of the robot (Plotnik and Longman 2000). One makes such a plot for a range of integer phase leads, and picks the one that keeps the plot less than one up to the highest frequency. Of course, one can also optimize the learning gain by plotting for different values.

![Figure 12. Nyquist plot of the closed-loop timing-belt drive system.](image)

![Figure 13. Nyquist plot for linear phase lead.](image)
11. Producing a frequency cutoff

11.1. Physical considerations affecting the choice of cutoff frequency

Besides trying to satisfy (33), there are two other issues that may suggest the use of a cutoff.

11.1.1. Acknowledging hardware limitations. When the phase cancellation repetitive control law was applied to the timing belt drive system (Hsin et al. 1997a), the error frequency content shown in figure 11 was produced, which is essentially perfect up to the Nyquist frequency of 500 Hz. The output was a nearly perfect constant shaft velocity, but the motor was making very fast corrections to cancel all the high frequency peaks in figure 5. In doing so, the hardware was making so much noise that we were worried about damaging it. The very large peak at 240 Hz corresponds to the tooth meshing frequency of the faster rotating belt, and the input motor is correcting for these small fluctuations and eliminating their effect on the output. This is being done very far above the bandwidth of the feedback controller. A command that cancels a sinusoid at this frequency in the output must be 11 times as large as the signal being cancelled. Clearly the hardware was being worked very hard. The conclusion is that practical hardware considerations will often demand that you not try to fix errors at particularly high frequencies. In addition, if the error above some frequency is so small that one is not interested in fixing it, one can use a cutoff at the frequency to be conside- 
rate of the hardware.

11.1.2. Control system robustness. The approach to designing ILC/RC in §§ 8–10 makes direct use of experimentally obtained frequency response information. If this information is accurate, then the cutoff can be chosen as in the previous section using equation (33). There will of course be some scatter in the data, see figures 12 and 13. If the scatter is large enough to make uncertain the frequency at which the curve goes out of the unit circle, then one must lower the cutoff frequency to be conservative, and to be assured of robustness of the law to the experimental errors. When experimentally tuning the ILC/RC law, growth of the error gives one data containing the frequencies that violate the monotonic decay condition. Hence they give you information about how much to lower the cutoff.

Note that designing the ILC/RC directly from the experimental frequency response information is a very important characteristic of the approach presented here. It may be tempting to try to design ILC/RC based on a mathematical model developed to fit data, but it is very likely that doing so will produce instability when applied in the real world. Nearly always the error in a model fit will be sufficiently large at high frequencies that equation (33) becomes violated in practice. Nearly always there are extra poles (sometimes called parasitic poles), or extra high frequency dynamics that one has not modelled, such as an extra vibration mode that you could not see in the data, or modelling a body as rigid when it has a slight flexibility, or modelling an amplifier as ideal when it actually has a first order pole at high frequency, etc. All it takes is one extra unmodelled pole and there is an extra 90° of phase lag at high frequency. This is normally sufficient to send the plot outside the unit circle, and the ILC or RC unstable. By relying directly on the data model, we eliminate this sensitivity problem.

11.2. Methods of producing a cutoff

It is important that the method of producing the cutoff not introduce its own phase lag altering the needed cutoff frequency. But no causal filter will produce a cutoff without introducing phase lag. The following two filtering options produce zero phase change, but they are implemented in a batch mode.

11.2.1. Zero-phase IIR low pass filtering. In Elci et al. (1994a,b) and Hsin et al. (1997a) zero-phase infinite impulse response (IIR) low-pass filtering is used. A high order Butterworth filter is chosen to create something close to an ideal filter, passing frequencies below the cut-off without much amplitude change, and then decaying fast after the cut-off. One can of course consider other similar filters. However, like any causal filter it produces phase lags. To get zero phase, first filter the data forward in time, and then apply the filter again to the results but filtering backward in time. This produces twice the attenuation of a single application of the filter and cancels the phase change produced in the first application. This is a very effective filter approach for our purpose.

There are some issues in applying the method. One wants the result to be free of any filter transients, in order to realize the desired filter steady state frequency...
response characteristics. There are filter initial conditions for both applications, first at the start of the data set, and in the second application they correspond to the final time in the data. In ILC this issue is addressed by extending the signal being filtered at both ends of the trajectory in an appropriate way, using a sufficient number of points (Plotnik and Longman 2000). For example one can simply repeat the endpoint for many time steps, or pad with zeros. One can also do an extension that tries to maintain continuity and continuity of the first derivative across the endpoints, by suitable reflections of points near the end. Note that using the default extensions in MATLAB may not be sufficient, and the user should be prepared to perform his own extensions. In repetitive control, this extension problem is solved by using more real data, as described below.

The fact that this is a non-causal filter that cannot be performed in real time, is not a problem in ILC where one can perform the computation between repetitions, or apply the result to the next repetition when ready. In RC one can easily distribute the filtering computations through the real-time computations, and then a batch update of the RC action is made when the computation is ready. One approach is as follows (Hsin et al. 1997 b). Start filtering in real-time as the data arrives, filtering from the beginning of one period and continuing through three periods. Then continue filtering in real time, but filtering the previous stored result backward until finished at the end of the 6th repetition. Starting with the 7th repetition, apply the zero-phase filtered result corresponding to the middle of the three periods filtered. The other two periods just served to allow decay of the filter transients both forward and backward (the smoothing procedures in Chen and Longman (1999) can also be used to give improved results). If the compensator works well, and if \( \Phi(z) \) contains a low pass filter as discussed in §11.4, then the time between batch filtering updates can be made long, and between these updates, one continues to apply the RC law. For example, in one application on the timing belt drive tested using a 12th order filter in \( \Phi(z) \), using a zero phase IIR filter every 2000 repetitions would be sufficient (Hsin et al. 1997 b).

### 11.2.2. Zero-phase cliff filter.

A batch computation that in theory forms an ideal cut-off is based on discrete Fourier transforms. Transform the signal into its frequency components, eliminate those above the cutoff, and then transform back. Call this a cliff filter, since theoretically it produces a perfect cliff at the desired frequency. Handling leakage effects is done the same way as transients in the IIR filtering above (Plotnik and Longman 2000), either extending the endpoints of the signal in the case of ILC, or using more data in the case of RC. There can be advantages in ILC to be in the frequency domain, speeding up the computation of a convolution sum ILC law as a product in the transformed space (Lee-Glauser et al. 1996).

### 11.2.3. Other options.

There are some other options of how to prevent difficulty from frequencies that go outside the unit circle. One is to stabilize by quantization, limiting the number of digits used in the ILC/RC updates (Hsin et al. 1997 b,c), but a filter of the above type can still be helpful to cut out components above the cutoff that accumulated from transient effects (Chen and Longman 1999). Another alternative is to use experimentally determined matched basis functions (Oh et al. 1997, Wen et al. 1998, 2000, Frueh and Phan 1999), where the choice of basis functions implies a cutoff. In the ILC problem this approach prevents divergence, and in RC with slow enough learning it also prevents divergence (Longman et al. 2000). A third alternative, with ILC laws such as phase cancellation, one can use on-line identification of the phase lag. When the phase lag information used in the learning law is sufficiently wrong to cause growth of error, it also produces precisely the data needed to know how to fix the problem (Elci et al. 1994 c, Longman and Wang 1996).

### 11.3. What signal to cutoff/predicting the steady state error levels

Initially, one might simply think to use the cutoff on the error signal, cutting out the unwanted frequency components before the ILC/RC law sees the error. If there are no components left outside the unit circle, then they cannot grow. However, in the case of a zero-phase IIR filter, the filter does not produce a perfect cutoff, but lets a small amount of signal above the cutoff through. Eventually this will accumulate, causing the tracking error to grow. In applications, the learning process is used for only eight or 12 repetitions to find what command to apply to a control system to give good tracking performance. In this case long term stabilization is not an issue and filtering only the command is appropriate (Longman and Huang 1994).

When the learning is left on, one must be sure that there is no accumulation of signal outside the unit circle. There are two options (Longman and Songchon 1999). One is to apply filter \( F \) to the total signal that is given as a command to the feedback controller. For simplicity, assume that this filter is applied to every repetition, as one would do in ILC with sufficient time between repetitions for the computations. The command for each repetition is given by

\[
U_{j+1}(z) = F[U_j(z) + z^2 \Phi(z)E_j(z)]; \quad U_0(z) = Y^*(z)
\]

The second option is to leave out the desired trajectory from the repeated filtering process and only filter the
learning or repetitive control modifications to the command that is produced by the ILC/RC learning process
\[
U_j(z) = Y^*(z) + U_{L,j}(z) \\
U_{L,j+1}(z) = F[U_{L,j}(z) + \Phi(z)E_j(z)]; \quad \{35\}
\]
\[U_0(z) = 0\]

In the case of a cliff filter, \( F \) is unity up to the cutoff frequency and zero above the cutoff. For the zero phase Butterworth filter, suppose the magnitude versus frequency function of the chosen order filter is \( M_F(\omega) \). When applied in zero phase form, this magnitude change appears twice, and there is zero phase, making the frequency transfer function of the filter equal to \( F(\exp(i\omega T)) = M_F(\omega) \). Now develop the convergence condition and the final error level for each option.

**Option 1:** The error at repetition \( j \) is equal to 
\[ E_j(z) = Y^*(z) - G(z)U_j(z) - V(z), \]
and we wish to combine this with (34) and eliminate \( U \) dependence so that we have an equation for the error in terms of the two inputs, \( Y^*(z) \) and \( W(z) \). To do this, write the difference \( E_j(z) - FE_{j-1}(z) \) in order to create a term \( U_j(z) - FU_{j-1}(z) \) that can be replaced by \( F\Phi(z)E_{j-1}(z) \) according to (34). The result is
\[ E_j(z) - F(1-G(z)\Phi(z))E_{j-1}(z) = (1-F)(Y^*(z) - V(z)) \]  
(36)
The right-hand side represents a forcing function, and the left-hand side represents a difference equation. If there were no cutoff, the right-hand side would be zero, but the left-hand side homogeneous equation would be unstable. By using a cutoff, one produces stability for the left-hand side, at the expense of leaving some forcing function on the right. In particular, for an ideal filter, the right is zero below the cutoff and passes the components above the cutoff unchanged. And this remaining part produces a steady state particular solution. Examining the homogeneous equation, it can be written as
\[ E_j(z) = [F(1-G(z)\Phi(z))]E_{j-1}(z) \]
(37)
Hence, the monotonic decay condition for the learning transients when there is a cutoff filter becomes
\[ |M_F^2(\omega)[1-G(e^{i\omega T})e^{r\omega T}\Phi(e^{i\omega T})]| < 1 \]  
(38)
for all \( \omega \) up to Nyquist frequency. One picks the order of the Butterworth filter and its cutoff in order to satisfy this condition. Equation (38) can be interpreted as saying, if the growth of signal components outside the unit circle in \( 1-G(e^{i\omega T})e^{r\omega T}\Phi(e^{i\omega T}) \) times the attenuation from the Butterworth filter \( M_F^2(\omega) \) is less than one, then convergence will result. When the filter is not applied to every repetition, the corresponding condition is that the growth between filtering will be less than the attenuation of the filter. Once (38) is satisfied, then we wish to know what the final error level will be, as a result of the forcing function. Once steady state error is reached, \( E_j(z) = E_{ss,j}(z) \) which we can call \( E_{ss}(z) \). The transfer function from \( Y^*(z) - V(z) \) to \( E_{ss}(z) \) with \( z = \exp(i\omega T) \) is
\[ H_1(\omega) = \frac{1 - M_F^2(\omega)}{1 - M_F^2(\omega)[1- G(e^{i\omega T})\Phi(e^{i\omega T})]} \]  
(39)
Multiply the amplitudes of the frequency components of \( Y^*(z) - V(z) \), determined above experimentally, by \( |H_1(\omega)| \) to get the amplitude of the steady state error at each frequency above the cutoff.

**Option 2:** The error at repetition \( j \) becomes 
\[ E_j(z) = (1-G(z))Y^*(z) - G(z)U_j(z) - V(z), \]
Use this in \( E_j(z) - FE_{j-1}(z) \) in order to create a term \( U_j(z) - FU_{j-1}(z) \) and replace it by \( F\Phi(z)E_{j-1}(z) \). The result analogous to (36) is
\[ E_j(z) - F(1-G(z)\Phi(z))E_{j-1}(z) \]
\[ = (1-F)(1-G(z))Y^*(z) - V(z) \]  
(40)
Examining this equation, the condition for monotonic decay of the learning transients is the same as (38). Also, the frequency transfer function from disturbance \( V(z) \) to its steady state particular solution error is again \( H_1(\omega) \). But the frequency transfer function from command to its steady state particular solution is different
\[ H_3(\omega) = \frac{[1-M_F^2(\omega)][1-G(e^{i\omega T})]}{1-M_F^2(\omega)[1-G(e^{i\omega T})\Phi(e^{i\omega T})]} \]  
(41)
Hence, if the point of the ILC/RC is to eliminate effects of repeating disturbances, with the command being zero or a constant, as is common in repetitive control problems, then there is no difference between the two options. On the other hand, when the main objective of the learning process is to eliminate tracking errors of the feedback controller following a command, which one is better depends on the dominant frequency range in the command. If the dominant frequency content of the command is in a range for which the ratio
\[ |H_3(\omega)|/|H_1(\omega)| = |1-G(e^{i\omega T})| \]  
(42)
is greater than unity, then it is best to filter the desired trajectory, Option 1. If the dominant error is in a frequency range where this plot is below one, then use Option 2. One applies (39) and (41) to the two inputs \( Y^*(z) \) and \( V(z) \) to predict the final error levels associated with the measured disturbance and the desired command, in order to decide which approach to use.

11.4. **Introducing a real time filtering cutoff**

In repetitive control one cannot apply the zero phase low pass filter every period. Between applications,
components outside the unit circle will grow. Thus, it is best if the Nyquist plot does not go far outside the unit circle so that this growth is small. This can be accomplished by using a causal IIR low pass filter in $\Phi(z)$. Although in repetitive control one has this extra motivation to keep the plot from extending far outside the unit circle, it can be desirable in learning control as well. For example, it may make it unnecessary to apply zero phase filtering except occasionally. It can even eliminate the need for zero phase filtering as described below.

The tuning of the linear phase lead ILC/RC design described in §10.2 is modified slightly in this new situation. The linear phase lead must be adjusted to not only compensate for the phase lag in $G(z)$ but that in $\Phi(z)$ as well. A simple design approach is to split $\gamma$ into two parts $\gamma_1 + \gamma_2$. Also write $\Phi(z) = \phi \Phi(z)$ where $\phi$ is the gain and $\Phi(z)$ is the Butterworth filter. Then adjust $\gamma_2$ just as before, to keep $z^{\gamma_2} \Phi(z)$ within the unit circle up to as high a frequency as possible. Having determined the needed cutoff, then pick $\Phi(z)$ as a high order Butterworth filter, and adjust $\gamma_1$ to keep $z^{\gamma_1} \Phi(z)$ inside the unit circle to as high a frequency as possible. Then combine the two to make the associated Nyquist plot. This gets one into the right range of values for the phase lead and the cutoff. Some adjustment of these values may be needed to make a final consistent design. An alternative to the Butterworth filtering is to use linear phase lead coupled with a zero-phase triangular or Bartlett window (Wang and Longman 1996, Hsin et al. 1997b). In this case the linear phase lead is only needed for $G(z)$. Triangular windows are much further from producing an ideal cutoff and have side lobes as well. But both approaches can be implemented in real time and can produce good results.

One can often make the amount by which the plot grows outside the unit circle very small, so that the maximum size of the error associated with all frequencies outside the circle is less than some very small number. By using a quantization of the error signal seen by the ILC/RC that is larger than this value, one produces long term stabilization, and no longer needs a zero phase filter. Hsin et al. (1997c) give the details of how to make ILC/RC that are stabilized based on quantization instead of zero phase filtering. Experiments are reported there where the quantization level needed for stabilization was below the number of digits carried by the digital to analogue, and analogue to digital converters—roughly seven digits accuracy. Long term stabilization was demonstrated both analytically and in an experiment run for 10 000 repetitions. Thus, it is possible that this design approach totally eliminates the need for zero phase filtering. However, it is still advantageous to apply zero phase filtering, because during the learning process there are transients that contain frequency components outside the unit circle. Although these do not grow to destabilize the learning, they influence the final error level. Use of smooth updating procedures helps to prevent such accumulations, and to get to a low final error level (Chen and Longman 1999), but zero phase filtering can filter out any accumulation from transients and produce the low final error level in a more robust manner.

12. Summary and conclusions

In classical control there are routine control laws: proportional (P), integral (I), PI, proportional plus derivative (PD), and PID. Using these laws involves tuning either one, two, or three parameters by rather routine procedures. Three types of ILC/RC laws have been described here, which again involve tuning only a small number of parameters. The method of tuning is more straightforward than for classical control. Hence, it is suggested that these laws can fill the same niche for ILC/RC as the P, I, PI, PD and PID do for classical control. Application of the methods will always decrease the tracking error, and normally result in very substantial reductions. Often one can approach the repeatability level of the hardware. In the case of the robot experiments a factor of 100 improvement in tracking error was obtained for a high speed manoeuvre using the first and simplest of these laws. Approaching a factor of 1000 is possible with the other versions.

The first law is integral control based with a frequency cutoff, the second adds a linear phase lead and the third adds a causal low pass filter to help produced a cutoff. The methods presume there is a functioning feedback control system. The first step in designing the ILC/RC is to perform a frequency response test on this system. In picking the ILC/RC gain $\phi$, it is easiest to simply pick a value in the range $1/4 \leq \phi \leq 1$. Lower values in this range favour smaller final error levels in the presence of noise, and slightly higher cutoff frequencies, while higher values favour faster learning. Then multiply the experimental $G(e^{\omega T})$ data by the frequency transfer function of the chosen ILC/RC law and plot the result, noting at what frequency the plot leaves the unit circle centred at +1. In the case of the linear phase lead ILC/RC law one makes this plot for a range of linear phase leads of one, two, three, etc. time steps and chooses the lead that keeps the plot inside the unit circle up to the highest frequency. This information allows you to choose the cutoff frequency. When using a causal low pass filter, there is some extra adjustment done in a similar manner, because the phase lead needs to compensate for the causal low pass filter as well. Then choose the method of cutoff, IIR (use (38)) or cliff, make the decision of what to filter using (39), (41) and (42), and the design is done. Equations (39), (41) and (42) predict the final error levels reached by the design.
If the cutoff was chosen too high, for example because of inaccurate data, then the signature of this condition is clear. When using a zero phase IIR, one monitors the RMS error to see if it goes above a threshold, violating the monotonic decay property. In the case of a cliff filter, one can actually see what frequency is growing and correct the cutoff accordingly. Therefore, one can monitor the system behaviour, and make corrections. Since the tuning is easy, one can create self tuning ILC/RC as discussed in Longman and Wirkander (1998) and Wirkander and Longman (2000).

If the compensation is good and a causal low pass filter is used, as in the third law, one may be able to apply the zero phase filtering only occasionally (e.g. every 2000 repetitions in some experiments on the timing belt would be sufficient). In fact, with some quantization introduced, either intentionally or for example in the A/D or D/A converters, it is possible to eliminate the zero phase filter altogether. But using a zero phase filter occasionally will normally produce better final error levels by eliminating some high frequency accumulation from transients in the data. There is another class of application where a zero phase cutoff is not needed. These are applications in which one simply applies the learning (without using a zero phase cutoff) until the error goes past a minimum, records the command that produced the minimum, and uses it without further update. All long term stability issues are thus bypassed. This can be particularly appropriate for robots on assembly lines, simply tuning the commands while setting up the line, and then using the commands thereafter.

There are two situations where the above design process may not yield the desired performance. The linear phase lead will not be able to get above a nearly undamped resonance in the system producing close to a 180° cliff in the phase. For this case, one can either introduce a cancellation of these lightly damped poles into the \( \Phi(\phi) \) or shift to the phase cancellation law (Elci et al. 1994c, Longman and Wang 1996, Hsin et al. 1997a). The second situation is when the desired trajectory in learning control is so short that steady state frequency response thinking is not representative, in which case one can shift to the ILC law based on a partial isometry (Jang and Longman 1996b). For RC the fix is easy, simply define the period to be some integer multiple of the actual period, so that the settling time is now short compared to the new period.

The field of iterative learning control started to develop fast in 1984, and repetitive control originated some years earlier. Although both fields offer substantial improvements in the performance of control systems doing repetitive tasks, they have seen relatively little application. We speculate on some of the reasons. In the case of ILC, the literature placed much of its emphasis on handling the non-linear equations for robotics, narrowing the possible users to a small set. It made control laws that required the robot manufacturer to replace his whole control approach. Concerning linear learning laws, the bad transients of integral control based learning made this simple approach impractical. For both learning and repetitive control there was the phenomenon of apparent initial convergence, and then apparent or actual divergence, perhaps after many repetitions. The possibility of such divergence is unsettling to an engineer who needs to deliver a working product. These difficulties prevented widespread use. This paper presents ILC/RC laws that address these issues. They are applicable to most any linear feedback control system. They are not specialized to robotics and do not require the feedback controller to have a specific form. The laws are simple and it is very easy to tune them. Since the difficulties that prevented wide application of ILC/RC are now addressed, the time is ripe for an explosion of applications in engineering practice.

References


OMATA, T., NAKANO, M., and INOUE, T., 1984, Applications of repetitive control method to multivariable systems. Transactions of SICE, 20, 795–800.


