On Quadratic Programming Based Iterative Learning Control for Systems with Actuator Saturation Constraints

Fei Gao and Richard W. Longman

Abstract When feedback control systems are given a commanded desired trajectory to perform, they produce a somewhat different trajectory. The concept of bandwidth is used to indicate what frequency components of the trajectory are executed reasonably well. Iterative Learning Control (ILC) iteratively changes the command, aiming to make the control system output match the desired output. The theory of linear ILC is reasonably well developed, but in hardware applications the nonlinear effects from hitting actuator saturation limits during the process of convergence of ILC could be detrimental to performance. Building on previous work by the authors and coworkers, this paper investigates the conversion of effective ILC laws into a quadratic cost optimization. And then it develops the modeling needed to impose actuator saturation constraints during the ILC learning process producing Quadratic Programming based ILC, or QP-ILC. The benefits and the need for ILC laws that acknowledge saturation constraints are investigated.

1 Introduction

Iterative learning control is a relatively new form of control theory that develops methods of iteratively adjusting the command to a feedback control system aiming to converge to that command that produces zero tracking error of a specific desired trajectory. References [1–3] develop various linear formulations. Reference [4] develops the supervector approach to mathematical modeling of ILC that is used here. There are some perhaps surprisingly strong mathematical convergence results for very general nonlinear systems, Refs. [5, 6], but they tend to use the simplest form of ILC that can have extremely bad transients [3] and may require that

F. Gao
Visiting Research Scholar, Columbia University, New York, NY 10027, USA

Doctoral Candidate, Tsinghua University, 100084 Beijing, China
e-mail: ambersro@gmail.com

R.W. Longman (✉)
Department of Mechanical Engineering, Columbia University, New York, NY 10027, USA
e-mail: rwl4@columbia.edu

H.G. Bock et al. (eds.), *Modeling, Simulation and Optimization of Complex Processes - HPSC 2012*, DOI 10.1007/978-3-319-09063-4_4

the system being controlled has the property that the output is instantaneously changed by a step change in the input [6]. Reference [7] develops methods of numerical computation for implementation of various effective linear ILC laws to nonlinear systems. Our objective here is to generalize methods of ILC to handle the specific kind of nonlinearity presented by inequality constraints on the actuators. The work follows on from [8] including generalization to form well posed inverse ILC problems, and is related to [9].

The present paper considers three effective linear ILC laws, a Euclidean norm contraction mapping ILC law which we refer to as the P transpose ILC law [10], the partial isometry ILC law [11], and the ILC law based on a quadratic cost penalty function on the transients, or changes in the control action from iteration to iteration [12, 13]. Reference [14] shows how one can create a unified formulation of each of these control laws by appropriate choices of the weights in the quadratic cost control. Most discrete time systems that come from a continuous time system fed by a zero order hold have an unstable inverse which implies that the control action needed for zero tracking error is an unstable function of time step. This difficulty can be addresses by the methods of Refs. [15–17].

This paper considers linear systems subject to inequality limits on actuators. It combines these approaches to form a quadratic cost minimization problem, formulating the needed equations to represent actuator saturation limits, thus forming a well posed quadratic programming problem [18, 19]. From the point of view of quadratic programming, these problems are small and easily solved for the updates to the commands made each iteration.

2 Two Classes of ILC Problems with Constraints

Time optimal control problems usually involve control actions that are bang bang, i.e. on the actuator constraint boundary. Fuel optimal control can also have actuators at their limits. One form of ILC considers the equations from actuator to output. In this case, it is reasonable to consider the relatively simple situation of an ILC problem where this input is saturated. Figure 1 illustrates this situation, which is referred to later as Problem 1. We use $G(s) = [a/(s+a)][\omega_0^2/(s^2 + 2\zeta\omega_0^2)]$ with $\omega_0 = 37$ and $\zeta = 0.5$, sampling at $T = 1/100$ s in numerical examples. ILC makes use of stored data from the previous run, and hence must it be digital and must use sampled data. Here we use a zero order hold. Converted to state variable form, the continuous time and discrete time modes are

Fig. 1 Problem 1, where ILC adjusts input that is subject to inequality constraints

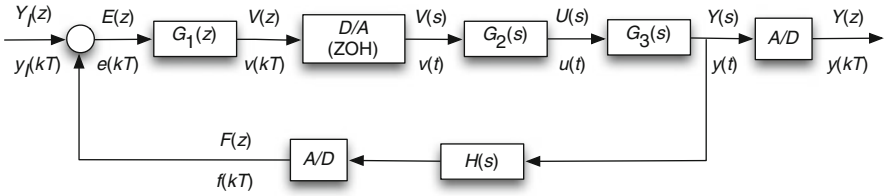

Fig. 2 Problem 2, a feedback control system with saturation limits on the actuator output

$$\begin{aligned} \dot{x}(t) &= A_c x(t) + B_c u(t) \quad ; \quad y(t) = Cx(t) \\ x((k+1)T) &= Ax(kT) + Bu(kT) \quad ; \quad y(kT) = Cx(kT) \\ A &= e^{A_c T} \quad ; \quad B = A_c^{-1}(A - I)B_c \end{aligned} \tag{1}$$

The more common situation has the ILC adjusting the command to a digital feedback control system, as in Fig. 2. The $G_1(z)$ represents the digital feedback control law while $G_2(s)$ represents the dynamics of an actuator whose output feeds the plant equation $G_3(s)$. An $H(s)$ is included because applications such as robotics often benefit from the use of rate feedback. The mathematical formulation of ILC subject to actuator output saturation is developed in general for hard saturation limits on $u(t)$.

Numerical examples for Problem 2 use $G_1(z) = K_1 = 12{,}050$, a proportional controller whose computation is considered instantaneous and therefore it does not produce a time step delay. The actuator dynamics are represented by $G_2(s) = 1/(s + a)$ with $a = 41.8$ while $G_3(s) = 1/(s^2 + 4s)$, $H(s) = 1 + 0.1268s$, and $T = 1/100$ s.

3 Several Effective Linear ILC Laws

3.1 General ILC Supervector Formulation

Consider a discrete time input output state space model with its convolution sum solution

$$\begin{aligned} x((k+1)T) &= Ax(kT) + Bu(kT) \quad ; \quad y(kT) = Cx(kT) + \upsilon(kT) \\ y(kT) &= CA^k x(0) + \sum_{i=0}^{k-1} CA^{k-i-1} Bu(iT) + \upsilon(kT) \end{aligned} \tag{2}$$

The $\upsilon(kT)$ represents any disturbance that repeats every run, and it is represented by its equivalent effect on the output. A one time step delay from a change in the input to the resulting change in the output is assumed. The mathematics can easily be altered to account for a different delay. Hence, for control action starting at time step zero, the desired trajectory $y^*(kT)$ is defined starting with time step one, and continuing to time step p, with associated error $e(kT) = y^*(kT) - y(kT)$. Following [4] we define symbols with underbars to represent the history of the associated variables for any run (subscript j will be applied to indicate iteration or run number j). Based on the one time step delay through the system, the entries in $\underline{u}$ start with time step 0 and go to $p-1$, and for $\underline{y}, \underline{y}^*, \underline{e}, \underline{\upsilon}$ start with time step 1 and go to p. Then

$$\underline{y} = P\underline{u} + \bar{A}x(0) + \underline{\upsilon} \quad ; \quad \underline{e} = -P\underline{u} + (\underline{y}^* - \bar{A}x(0) - \underline{\upsilon})$$
$$P = \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{p-1}B & CA^{p-2}B & \cdots & CB \end{bmatrix} \qquad \bar{A} = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^p \end{bmatrix} \tag{3}$$

3.2 General Linear ILC and Convergence

A general ILC law updates the input action according to

$$\underline{u}_{j+1} = \underline{u}_j + L\underline{e}_j \tag{4}$$

where L is a matrix of ILC gains. By taking the difference of the right hand equation in (3) for two successive iterations, one obtains the error convergence condition. By substituting the error equation in (3) into (4), one also obtains the control action convergence condition

$$\underline{e}_{j+1} = (I - PL)\underline{e}_j$$
$$\underline{u}_{j+1} = (I - LP)\underline{u}_j + L(\underline{y}^* - \bar{A}x(0) - \underline{\upsilon}) \quad ; \quad \Delta_{j+1}\underline{u} = (I - LP)\Delta_j\underline{u} \tag{5}$$

Defining $\Delta_j\underline{u} = \underline{u}_j - \underline{u}_\infty$ converts the second equation into the third. Convergence of the error requires that the spectral radius of $I - PL$ be less than unity, and a sufficient condition is that the maximum singular value is less than unity.

3.3 Euclidean Norm Contraction Mapping ILC Law (P Transpose Law)

This law picks $L = \phi P^T$ where ϕ is a positive gain to be chosen [10]. Write the singular value decomposition of P as $P = USV^T$. Note that P is lower triangular with nonzero elements on the diagonal, and therefore in theory is full rank with $S = diag(\sigma_1, \sigma_2, \ldots, \sigma_p)$ having all positive diagonal elements. It can however be badly ill conditioned, and this is accounted for below. Equations (5) can be rewritten for this law

$$U^T \underline{e}_{j+1} = (I - \phi S^2) U^T \underline{e}_j \quad ; \quad V^T \Delta_{j+1} \underline{u} = (I - \phi S^2) V^T \Delta_j \underline{u} \tag{6}$$

The Euclidean norms of $\underline{e}_j$ and $U^T \underline{e}_j$ are the same, and similarly for $\Delta_j \underline{u}$. Monotonic convergence is obtained for all initial error histories if and only if $1 - \phi\sigma_i^2$ is less than unity in magnitude for all i, or it converges if $0 < \phi < 2/\sigma_i^2$ for all i. If $1 - \phi\sigma_i^2$ is positive, then the corresponding component of the error will converge from the same side each update, and if it is negative this component of the error will alternate in sign from iteration to iteration.

3.4 Partial Isometry ILC Law

Set $L = \phi V U^T$ as in [11] and one obtains convergence to zero error as in Eq. (6) with $I - \phi S^2$ replaced by $I - \phi S$, with $0 < \phi < 2/\sigma_i$. This law learns faster for high frequency error components, but is somewhat less robust to model errors.

3.5 General Quadratic Cost ILC Law

Usually quadratic cost control implies a compromise between control effort and speed of decay of the transients. ILC wants zero error, but uses the quadratic cost compromise on the size of the control update from iteration to iteration, controlling the learning transients. The quadratic cost at iteration j that determines $\underline{u}_{j+1}$ minimizes

$$J_j = \underline{e}_{j+1}^T Q \underline{e}_{j+1} + \delta_{j+1} \underline{u}^T R \delta_{j+1} \underline{u} \quad ; \quad \delta_{j+1} \underline{u} = \underline{u}_{j+1} - \underline{u}_j \tag{7}$$

Normally, one asks that Q be positive semidefinite and R be positive definite. In this case, we want zero final tracking error so we want Q positive definite, and R need not be positive definite. Write the right hand equation in (3) for $j + 1$ and for j, and then (7) can be rewritten as

$$J_j = (\underline{e}_j - P\delta_{j+1}\underline{u})^T Q(\underline{e}_j - P\delta_{j+1}\underline{u}) + \delta_{j+1}\underline{u}^T R\delta_{j+1}\underline{u}$$
$$= \delta_{j+1}\underline{u}^T(P^T QP + R)\delta_{j+1}\underline{u} - 2\delta_{j+1}\underline{u}^T P^T Q\underline{e}_j + \underline{e}_j^T Q\underline{e}_j \tag{8}$$

Instead of R having to be positive definite, this quadratic cost law requires the Hessian $P^T QP + R$ to be positive definite, which actually allows for negative R. Setting the derivative with respect to $\delta_{j+1}\underline{u}$ to zero produces the ILC law

$$\underline{u}_{j+1} = \underline{u}_j + L\underline{e}_j = \underline{u}_j + (P^T QP + R)^{-1} P^T Q\underline{e}_j \tag{9}$$

3.6 Simple Quadratic Cost ILC

Often when applying quadratic cost control designs one does not have much guidance on how to pick the weight matrices, and one uses simple choices. Here we consider $Q = I$ because we want all error components to go to zero, and let $R = rI$ where r is a scalar gain. Then

$$L = V(rI + S^2)^{-1} SU^T$$
$$(U^T\underline{e}_{j+1}) = [I - S(rI + S^2)^{-1}S](U^T\underline{e}_j) \tag{10}$$
$$= diag\Big(1 - \frac{\sigma_i^2}{r + \sigma_i^2}\Big)(U^T\underline{e}_j) = diag\Big(\frac{r}{r + \sigma_i^2}\Big)(U^T\underline{e}_j)$$

In this case, each component of the error on the orthonormal basis vectors in U will converge monotonically without alternating sign when r is positive, but it is possible to have convergence for negative values provided $r > -\sigma_i^2$ for all i.

3.7 Defining Well Posed ILC Inverse Problems

Continuous time systems fed by a zero order hold can be converted to discrete time systems with identical output histories at sample times. For continuous time systems with pole excess of 3 or more and sufficiently fast sampling, this conversion introduces zeros outside the unit circle. This means that the inverse problem is unstable. References [15–17] address this problem, for example, by asking for zero error every other time step making a kind of generalized hold. We rewrite Eq. (3) as

$$\underline{e}_{D,j} = -P_D\underline{u}_j + \big(\underline{y}_D^* - \bar{A}_D x(0) - \underline{v}_D\big) \quad ; \quad \delta_{j+1}\underline{e}_D = -P_D\delta_{j+1}\underline{u} \tag{11}$$

Initially write the equation for all time steps at the faster sample rate used by the control input. Then delete whichever rows are associated with errors that are not to

be addresses. Note that this makes P a rectangular matrix. The General Quadratic Cost ILC Law generalizes to

$$\begin{aligned} J_j &= \underline{e}_{D,j+1}^T Q \underline{e}_{D,j+1} + \delta_{j+1} \underline{u}^T R \delta_{j+1} \underline{u} \\ \underline{u}_{j+1} &= \underline{u}_j + L_D \underline{e}_{D,j} \quad ; \quad L_D = (P_D^T Q P_D + R)^{-1} P_D^T Q \end{aligned} \tag{12}$$

Note that $P_D^T Q P_D$ is now positive semidefinite. The Euclidean Norm Contraction Mapping ILC law generalizes immediately to $L_D = \phi P_D^T$. The Partial Isometry Law needs the singular value decomposition of P_D denoted by $P_D = U \begin{bmatrix} S & 0 \end{bmatrix} \begin{bmatrix} V_1 & V_2 \end{bmatrix}^T = U S V_1^T$ making $L_D = \phi V_1 U^T$.

4 Quadratic Cost Versions of These Effective ILC Laws

Equation (7) is the general version of quadratic cost ILC. By setting $Q = I$ and $R = rI$ the general version reduces to the Simple Quadratic Cost ILC law. However, the other two ILC law presented can also be produced from the general quadratic cost function. The Euclidean Norm Contraction Mapping Law $L = \phi P^T$ is produced when one sets $Q = \phi I$ and $R = I - \phi P^T P$ and substitutes into the L of Eq. (9). The partial isometry law is obtained when $R = VS(I - \phi S)V^T = P^T U(I - \phi S)V^T$ with the same $Q = \phi I$. We note that when substituted into L, one obtains $L = (VSV^T)^{-1} VSU^T \phi$. The inverse involved can be ill conditioned since S determines the condition number of P discussed above. The details of the computation of the updates using the quadratic cost formulation determine whether there is some additional difficulty when using this law in quadratic cost form.

One of the issues in the learning transients of ILC laws when there are inequality constraints is whether the iterations try to go beyond the actuator limit during the convergence process, and as a result perhaps the convergence process might fail. We note that the Simplified Quadratic Cost ILC Law in Eq. (10), when r is picked larger than zero, will have every component of the error projected onto the unit vectors of U converging to zero without changing sign. Thus if the initial trajectory starts with each of these components smaller than those needed for zero error, and the desired trajectory is feasible, i.e. it does not require actuator output beyond the actuator limits, then one expects convergence without difficulty using the learning law without regard to the inequality constraints. Using r that is negative seems counter intuitive, and furthermore how negative it can be is less than minus the smallest singular value which is often a very small number. When using the P transpose law, picking ϕ so that $0 < 1 - \phi\sigma_i^2 < 1$ for all i, will similarly ensure monotonic approach of the same sign to zero error for each of the error components on the unit vectors of U. Any i for which $-1 < 1 - \phi\sigma_i^2 < 0$ will alternate the sign of this component of the error each iteration. If the desired trajectory is feasible but goes near the actuator limit, the alternating sign of error components could easily

make the control law ask to go beyond the limit. For the partial isometry law, the corresponding conditions are $0 < 1 - \phi\sigma_i$ and $-1 < 1 - \phi\sigma_i < 0$.

To address the ill conditioning problem discussed above, and also convert to the P transpose and partial isometry laws to quadratic cost ILC form we pick weights in cost function (12) as follows. For P transpose, simply use $Q = \phi I$ and $R = I - \phi P_D^T P_D$. For the partial isometry law one again uses $Q = \phi I$, and the R matrix becomes more complicated

$$R = V \begin{bmatrix} S(I - \phi S) & 0 \\ 0 & I \end{bmatrix} V^T \tag{13}$$

Substitution into Eq. (12) produces the needed $L_D = \phi V_1 U^T$.

5 The Quadratic Programming Problem for Problem 1

Having produced quadratic cost versions of each of the above control laws, we are ready to impose inequality constraints on the actuator by formulating the update for each iteration as a quadratic programming (QP) problem:

$$\text{min: } J = \frac{1}{2} z^T H z + g^T z \quad \text{subject to: } A_1 z \le Z_1 \text{ and } A_2 z = Z_2 \tag{14}$$

Using the general quadratic cost ILC law, which can represent any of the ILC laws discussed above by choice of Q and R, at each iteration we seek to minimize J_j of Eq. (7). The equality constraint $A_2 z = Z_2$ could be used for the dynamics $\delta_{j+1}\underline{e} = -P\delta_{j+1}\underline{u}$, but we can simply substitute analytically to obtain the cost equation (8) with $H = \frac{1}{2}(R + P^T Q P)$ and $g^T = \underline{e}_j^T Q P$. In Problem 1, the input is subject to inequality constraints $\underline{u}_{min} \le \underline{u} \le \underline{u}_{max}$ which is to be interpreted component by component. These constraints are imposed with A_1, Z_1 using $\delta_{j+1}\underline{u} \le \underline{u}_{max} - \underline{u}_j$ and $-\delta_{j+1}\underline{u} \le -(\underline{u}_{min} - \underline{u}_j)$.

6 Formulating the Quadratic Programming Problem for Problem 2

6.1 Closed Loop Dynamics for Problem 2

Generation of the QP version of Problem 2 requires some effort to formulate the inequality constraints. We assume that the hardware has a hard constraint at its limit. This time the constraint is on the output of a continuous time transfer function that may represent the actuator, $u_{min} \le u(t) \le u_{max}$. For simplicity, we formulate the

problem asking to satisfy the constraints at the sample times $u_{min} \leq u(kT) \leq u_{max}$, and ignore any issues associated with violation of constraints between sample times. Referring to Fig. 2, we can write equations going around the loop. For the controller

$$\begin{aligned} e(kT) &= y_I(kT) - f(kT) \\ f(kT) &= y(kT) + K\dot{y}(kT) \\ x_d((k+1)T) &= A_d x_d(kT) + B_d e(kT) \\ v(kT) &= C_d x_d(kT) + D_d e(kT) \end{aligned} \tag{15}$$

We want to know not only the output $y(t)$ at least at the sample times, but also we want to monitor the value of $u(t)$ at the sample times, and for purposes of computing the feedback with $H(s) = 1 + Ks$ we want to know $\dot{y}(kT)$. The actuator and the plant equations in continuous time are

$$\begin{aligned} \dot{x}_a(t) &= A_{a,c}x_a(t) + B_{a,c}v(t) \quad ; \quad u(t) = C_a x_a(t) \\ \dot{x}_p(t) &= A_{p,c}x_p(t) + B_{p,c}u(t) = A_{p,c}x_p(t) + B_{p,c}C_a x_a(t) \quad ; \quad y(t) = C_p x_p(t) \\ \dot{y}(t) &= C_p\dot{x}_p(t) = C_p A_{p,c}x_p(t) + C_p B_{p,c}C_a x_a(t) \end{aligned} \tag{16}$$

One can combine these equations using a combined state $x_{ap} = \begin{bmatrix} x_a^T & x_p^T \end{bmatrix}^T$ and compute three output quantities

$$\begin{bmatrix} u(t) \\ y(t) \\ \dot{y}(t) \end{bmatrix} = \begin{bmatrix} C_a & 0 \\ 0 & C_p \\ C_p B_{p,c} C_a & C_p A_{p,c} \end{bmatrix} \begin{bmatrix} x_a(t) \\ x_p(t) \end{bmatrix} \tag{17}$$

The combined equations have input $v(t)$ coming from a zero order hold, and hence it can be converted to a difference equation giving these output quantities at the sample times without approximation. One can now combine these equations with (15) adding the controller state variable to those in (16) to form the closed loop state equations relating the command $y_{I,j}(kT)$ in iteration j to the resulting output $y_j(kT)$ and the actuator output $u_j(kT)$ which is subject to saturation

$$\begin{aligned} x_{CL,j}((k+1)T) &= A_{CL}x_{CL,j}(kT) + B_{CL}y_{I,j}(kT) \\ y_j(kT) &= C_{CL}x_{CL,j}(kT) \\ u_j(kT) &= C_u x_{CL,j}(kT) \end{aligned} \tag{18}$$

6.2 Dynamics When the Actuator Is Saturated

When the actuator is at a saturation limit u_{sat}, then when simulating, the feedback loop is only needed to determine when the actuator leaves its saturated value. The output and its derivative are given by

$$\begin{gathered} x_p((k+1)T)=A_p x_p(kT)+B_p u_{sat}\ ;\ A_p=exp(A_{p,c}T)\ ;\ B_p=A_{p,c}^{-1}(A_p-I)B_{p,c} \\ y(kT)=C_p x_p(kT)\quad ;\quad \dot{y}(kT)=C_p A_p x_p(kT) \end{gathered} \tag{19}$$

To determine in simulation when the actuator leaves saturation, one uses these values in Eqs. (15) to determine $v(kT)$, and feeds this into the discrete time version of the first row equations of (16) in order to monitor the value of $u(kT)$.

6.3 The QP Problem for Problem 2

Form two different P matrices, P_{CL} making use of system matrices A_{CL}, B_{CL}, C_{CL}, and the second P_u using A_{CL}, B_{CL}, C_u. For simplicity we ignore the repeating disturbance term so that

$$\begin{aligned} \underline{y}_j &= P_{CL}\underline{y}_{I,j}+\bar{A}_{CL}x_{CL}(0) \\ \underline{e}_j &= -P_{CL}\underline{y}_{I,j}+(\underline{y}^*-\bar{A}_{CL}x_{CL}(0)) \\ \underline{u}_j &= P_u\underline{y}_{I,j}+\bar{A}_u x_{CL}(0) \\ \delta_{j+1}\underline{u} &= P_u\delta_{j+1}\underline{y}_I \end{aligned} \tag{20}$$

where $\bar{A}_{CL}$ and $\bar{A}_u$ are formed as in Eq. (3) using A_{CL} and C_{CL} or C_u respectively. Then the QP problem at iteration j is to minimize the following quadratic cost subject to the following inequality constraints

$$\begin{gathered} J_j=\delta_{j+1}\underline{y}_I^T(P_{CL}^T Q P_{CL}+R)\delta_{j+1}\underline{y}_I-2\delta_{j+1}\underline{y}_I^T P_{CL}^T Q\underline{e}_j+\underline{e}_j^T Q\underline{e}_j \\ \underline{u}_{min}\le \underline{u}_j+P_u\delta_{j+1}\underline{y}_I\le \underline{u}_{max} \end{gathered} \tag{21}$$

7 Numerical Study of QP Based ILC

To create a desired trajectory that rides the actuator or input limit we pick $u(t)=sat\{0.5[1-\cos(2\pi t)]\}$, saturated at the value 0.7. Then we compute the output and use it as the desired trajectory. In some applications the inequality constraint

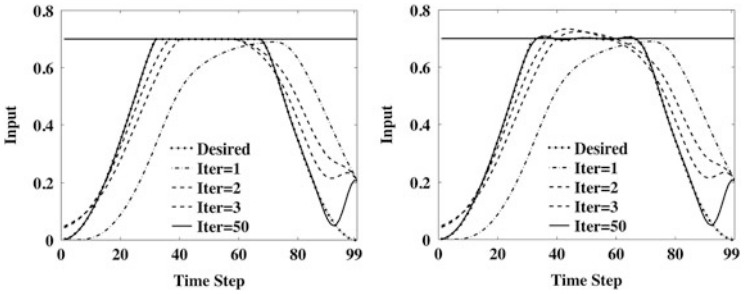

Fig. 3 Problem 1, P transpose ILC law with $\phi = 1$. *Left*: hardware imposes the limit. *Right*: hardware allows violation of limit

is a hard constraint that cannot be violated. For example, sometimes when a DC motor is used in a control system, it has a voltage limiter on it that imposes a hard constraint. Other times, the manufacturer lists a limit which can be based on such things as heating constraint in sustained operation. In this case, violating the constraint temporarily during the learning process can be acceptable. Here we examine the behavior of ILC laws in three cases: (1) When the law and the hardware are allowed to violate the limit temporarily. (2) When the law is allowed to ask for signals that violate the limit, but the hardware imposes the limit. (3) When the corresponding QP based ILC law incorporates the limits in the updates each iteration.

Figure 3 shows the convergence to the needed $u(kT)$ using a P transpose ILC law when the hardware does not allow violation of the constraint, and when it does. The root mean square (RMS) of the tracking error for each iteration is essentially identical, but the former learns slightly faster. It is interesting to note that although the mathematics indicates that all time steps converge to zero error, the last time step in these plots of the $u(kT)$ is converging very slowly. This phenomenon has now been studied in Ref. [20].

An interesting example was run using the P transpose ILC law with a gain of $\phi = 2.3$ which is above the stability limit of 2.2. As one might expect, the RMS error when violation was allowed went unstable. But with the hardware imposing the limit, convergence approaching zero error occurred, i.e. the hardware limit stabilized an unstable ILC.

Figure 4 illustrates the RMS error performance as ILC iterations progress for the partial isometry ILC law and the simple quadratic cost ILC law. The dashed curves in each case show the convergence when the ILC law could violate the constraint but the hardware imposed the constraint. The corresponding curves when the hardware allowed violation of the constraints looked very similar, except that the somewhat quick changes in slope in the figures disappear. The solid curves are the RMS errors for the corresponding QP algorithms that make use of the limits in the updates computed. Note that in both cases, the QP algorithm eventually outperforms the

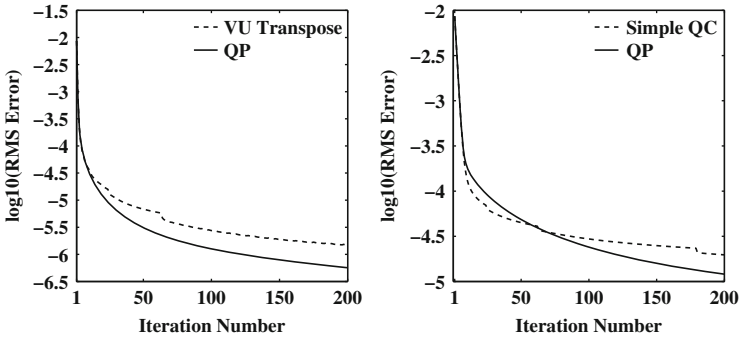

Fig. 4 Problem 2. *Left*: UV^T ILC with $\phi = 1$, *Right*: simple quadratic cost ILC, hardware imposes limit, $r = 1$. Versus QP

other algorithms. Note that the VU^T law reaches lower error levels since it converges linearly instead of quadratically in small errors.

References

1. Bien, Z., Xu, J.X. (eds.): Iterative learning control: analysis, design, integration and applications. Kluwer Academic, Boston (1998)
2. Moore, K., Xu, J.X. (guest eds.): Special issue on iterative learning control. Int. J. Control **73**(10) (2000)
3. Longman, R.W.: Iterative learning control and repetitive control for engineering practice. Int. J. Control **73**(10), 930–954 (2000)
4. Phan, M., Longman, R.W.: A mathematical theory of learning control for linear discrete multivariable systems. In: Proceedings of the AIAA/AAS Astrodynamics Conference, Minneapolis, pp. 740–746 (1988)
5. Longman, R.W., Chang, C.K., Phan, M.Q.: Discrete time learning control in nonlinear systems. In: A Collection of Technical Papers, AIAA/AAS Astrodynamics Specialist Conference, Hilton Head, pp. 501–511 (1992)
6. Xu, J.X., Tan, Y.: Linear and Nonlinear Iterative Learning Control. Springer, Berlin/New York (2003)
7. Longman, R.W., Mombaur, K.D.: Implementing linear iterative learning control laws in nonlinear systems. Adv. Astronaut. Sci. **130**, 303–324 (2008)
8. Longman, R.W., Mombaur, K.D., Panomruttanarug, B.: Designing iterative learning control subject to actuator limitations using QP methods. In: Proceedings of the AIAA/AAS Astrodynamics Specialist Conference, Hawaii (2008)
9. Mishra, S., Topcu, U., Tomizuka, M.: Optimization-based constrained iterative learning control. IEEE Trans. Control Syst. Technol. **19**(6), 1613–1621 (2011)
10. Jang, H.S., Longman, R.W.: A new learning control law with monotonic decay of the tracking error norm. In: Proceedings of the Thirty-Second Annual Allerton Conference on Communication, Control, and Computing, Monticello, pp. 314–323 (1994)
11. Jang, H.S., Longman, R.W.: Design of digital learning controllers using a partial isometry. Adv. Astronaut. Sci. **93**, 137–152 (1996)

12. Phan, M.Q., Frueh, J.A.: System identification and learning control, chapter 15. In: Bien, Z., Xu, J.X. (eds.) Iterative Learning Control: Analysis, Design, Integration, and Applications, pp. 285–306. Kluwer Academic, Norwell (1998)
13. Owens, D.H., Amann, N.: Norm-optimal iterative learning control. Internal Report Series of the Centre for Systems and Control Engineering, University of Exeter (1994)
14. Bao, J., Longman, R.W.: Unification and robustification of iterative learning control laws. Adv. Astronaut. Sci. **136**, 727–745 (2010)
15. Li, Y., Longman, R.W.: Characterizing and addressing the instability of the control action in iterative learning control. Adv. Astronaut. Sci. **136**, 1967–1985 (2010)
16. Li, Y., Longman, R.W.: Addressing problems of instability in intersample error in iterative learning control. Adv. Astronaut. Sci. **129**, 1571–1591 (2008)
17. Li, T., Longman, R.W., Shi, Y.: Stabilizing intersample error in iterative learning control using multiple zero order holds each time step. Adv. Astronaut. Sci. **142**, 2965–2980 (2012)
18. Coleman, T.F., Li, Y.: A reflective Newton method for minimizing a quadratic function subject to bounds on some of the variables. SIAM J. Optim. **6**(4), 1040–1058 (1996)
19. Gill, P.E., Murray, W., Wright, M.H.: Practical Optimization. Academic, London (1981)
20. Gao, F., Longman, R.W.: Examining the learning rate in iterative learning control near the end of the desired trajectory. Adv. Astronaut. Sci. **148**, 2019–2037 (2013)