UKACC International Conference on Control 2012
Cardiff, UK, 3-5 September 2012

A Convergence Analysis of D-ILC Algorithm

Inam Ul Hasan Shaikh
(*Student Member IEEE*)
Control Systems Centre
The University of Manchester
Manchester, UK
Inamulhasan.Shaikh@postgrad.manchester.ac.uk

Martin Brown
Control Systems Centre
The University of Manchester
Manchester, UK
Martin.Brown@manchester.ac.uk

***Abstract*—ILC is an emerging technique for learning control. The D-ILC algorithm is the generic ILC scheme which captures the error trend of a batch to update the control input for the next batch or batches. The 2-dimensional nature requires in-depth convergence analysis of the algorithm. This paper addresses these issues in detail. This paper deals with the convergence properties of ILC algorithms with emphasis on control input. Discrete-time linear state space representation of a linear time-invariant system has been considered along with usual assumptions which ensure D-type ILC algorithm converges in terms of output error. The convergence for control input sequence is investigated up to component level.**

Keywords: Iterative Learning Control; Rate of Convergence;

I. INTRODUCTION

Iterative Learning Control (ILC) is one of the Intelligent Control schemes. The transient performance is improved for systems which operate in a repetitive manner. The repetitions occur after fixed intervals of time. ILC has achieved better performance of control systems especially when dealing with uncertain/stochastic systems[1]. The concept of learning through repeated trials evolved in late 1970s for improving the motion control of mechanical arms [2, 3]. The D-type algorithms for Iterative Learning Control for linear time-varying systems with application to robotic manipulators were developed [4-6]. However, the D-type ILC suffered with the problem of differentiation of high frequency noise. Later, the P-type ILC improved upon by using only the error instead of its derivative [7].

Current Iteration Tracking Error (CITE) was introduced which formulated ILC in line with feedback control paradigm and helped to overcome large overshoot thus convergence could be accelerated [8]. The discrete-time version of D-type ILC was formulated for MIMO linear systems which possessed global robustness against state disturbances, measurement noise and re-initialisation error at the beginning of each iteration[9]. The ILC law has been employed for non-linear time varying systems having affine input and linear output. Uniform convergence of input and state was achieved when there were no disturbances [10].

In real world applications, the plants are usually non-linear with higher order dynamics and can not be modelled accurately. The mathematical model is linearised around a known equilibrium point for LTI systems. It then becomes easier to design the controllers. However, the model uncertainties restrict the controller from achieving optimal solutions.

Iterative Learning Controllers provide an adaptive solution. These controllers utilize the error information of each batch/iteration and update the control input accordingly for the next batch or batches. The control input signal using ILC converges to the desired value of the control input which is the inverse solution. Hence, it is also termed as Iteration Inversion Process.

The error information at each time instance can be used in a variety of ways to generate the update for control input. Hence, there are D, PD, PI, PID, Gradient based etc. many types of ILC controllers in which the output error, its derivative, integral or some combination of these is added to the current control input to generate input for the same time instant in next batch [7, 9-19]. The rate of convergence up to component level inside batches has not been covered in earlier works. This has been carried out in this paper.

II. DISCRETE-TIME SYSTEM WITH ILC

The discrete-time LTI system having one relative degree is considered as follows:

$$\begin{aligned} x(i+1,k) &= A\,x(i,k) + B\,u(i,k) \\ y(i,k) &= C\,x(i,k) \end{aligned} \tag{1}$$

where k denotes the Batch/iteration/trial number having M number of samples in each trial, $i \in [1, M]$ is the time index or sample number during each batch, state vector $x \in \mathbb{R}^n$, input $u(i,k) \in \mathbb{R}^r$ and output of the system is $y(i,k) \in \mathbb{R}^p$. A, B and C are the real-valued state, input and output matrices respectively, having appropriate dimensions. The initial conditions have been selected as follows:

- $x(1,k) = x_0$ is same at the start of each batch. Hence zero initial error is maintained.
- $u(i,1) = u_0(i)$ is the control input vector for first batch which may be externally specified or left to be zero [9, 10, 19, 20].

Here the control input for the first batch is assumed zero.

A. D-ILC Control Input Update

Control input is updated using D-ILC as follows:

$$u(i,k+1) = u(i,k) + K_d\{e(i+1,k) - e(i,k)\} \quad (2)$$

where K_d is the real-valued learning gain matrix and the derivative of error $\dot{e}(i,k)$ has been approximated using forward difference:

$$\dot{e}(i,k) = e(i+1,k) - e(i,k) \quad (3)$$

where $e(i,k)$ is the error between the desired/reference and the actual outputs:

$$e(i,k) = y^*(i) - y(i,k) \quad \text{for } 1 \le i \le M \quad (4)$$

Furthermore, using the zero initial error assumption we have $e(1,k) = 0$ i.e., $y^*(1) = y(1,k)$.

Thus, the objective of the D-type ILC algorithm is to find the sequence $u(i,k)$ so that:

$$\lim_{k\to\infty} u(i,k) = u^*(i) \quad \text{for } \forall i = 1,2,\ldots,M \quad (5)$$

For a linear system, the convergence of input sequence corresponds to the convergence of the output sequence:

$$\lim_{k\to\infty} y(i,k) = y^*(i) \quad \text{for } \forall i = 2,3,\ldots,M+1 \quad (6)$$

B. Convergence of Control Input Sequence using D-ILC

In this section, the component-level relationship for $u(i,k+1)$ at $(k+1)^{\text{th}}$ batch is presented in terms of static and dynamic components as well as the control input $u(i,k)$ from previous batch *k*. Convergence of control input $u(i,k)$ approaching the desired input $u^*(i)$ is investigated. So that the desired output sequence $y^*(i)$ is generated when sequence $u^*(i)$ is applied to the system:

$$y^*(i) = G(z)u^*(i) = \left\{C(zI-A)^{-1}B + D\right\}u^*(i) \quad (7)$$

The convergence condition for control input and rate of convergence for individual components of the control input have been derived, where bounds for the convergence rates have been formulated as well.

1) Batch to Batch Control Input Sequence

The D-type ILC algorithm generates the batch to batch control input sequence, i.e. from $1,2,\ldots,k,k+1,\ldots$ for each time index i. In this section a recurrence relationship is derived to perform a convergence analysis on $u(i,k)$. The control input sequences for individual time indices are derived using (2) as follows:

$$u(1,k+1) = u(1,k) + K_d\{e(2,k) - e(1,k)\} \quad (8)$$

Using the zero initial error, i.e., $e(1,k) = 0$, we get:

$$\begin{aligned} u(1,k+1) &= u(1,k) + K_d\{e(2,k)\} \\ &= u(1,k) + K_d\{y^*(2)\} - K_d C \begin{Bmatrix} Ax(1,k) \\ +Bu(1,k) \end{Bmatrix} \\ &= \left(I - K_d CB\right)u(1,k) + K_d\{y^*(2)\} - K_d CAx_0 \end{aligned} \quad (9)$$

It can be further expressed as follows:

$$\begin{aligned} u(1,k+1) &= \left(I - K_d CB\right)u(2,k) + K_d\left[y^*(3) - y^*(2)\right] \\ &\quad + K_d C(I-A)x(2,k) \end{aligned} \quad (10)$$

The sequence in (10) can be generalized as follows:

$$u(i,k+1) = \begin{pmatrix} \left(I - K_d CB\right)u(i,k) + K_d \begin{bmatrix} y^*(i+1) \\ -y^*(i) \end{bmatrix} \\ + K_d C(I-A)x(i,k) \end{pmatrix} \quad (11)$$

Solution for (11) is expressed as a recurrence relation:

$$u(i,k) = \left(I - K_d CB\right)^{k-1} u(i,1) + \sum_{j=1}^{k-1} \left(I - K_d CB\right)^{k-j-1} \begin{bmatrix} K_d\{y^*(i+1) - y^*(i)\} \\ +K_d C(I-A)x(i,k) \end{bmatrix} \quad (12)$$

For zero control input at first batch, (i.e. $u(i,1) = 0$), (12) can be reduced as follows:

$$u(i,k) = \sum_{j=1}^{k-1} \left(I - K_d CB\right)^{k-j-1} \begin{bmatrix} K_d\{y^*(i+1) - y^*(i)\} \\ +K_d C(I-A)x(i,k) \end{bmatrix} \quad (13)$$

As batch number k increases, the control input components achieve convergence one after the other with increasing time index i inside a batch. From re-writing the sequences in (11) to show the dependency of $u(i,k+1)$ on $u(i,k)$ from previous batch including all the other previous control input components $u(i-1,k)$, $u(i-2,k),\ldots,u(1,k)$ occurring in the same batch as follows:

$$\begin{aligned} u(i,k+1) &= \left(I - K_d CB\right)u(i,k) + K_d C \begin{pmatrix} (I-A) \\ Ax(i-1,k) \end{pmatrix} \\ &\quad + K_d C \begin{pmatrix} (I-A) \\ Bu(i-1,k) \end{pmatrix} + K_d\left[y^*(i+1) - y^*(i)\right] \end{aligned} \quad (14)$$

On further simplification, we have:

$$u(i,k+1) = \left(I - K_dCB\right) u(i,k) + K_dC\,(I-A)\,Bu(i-1,k) + \cdots + K_dC\,(I-A)\,A^{i-2}Bu(1,k) + K_dC\,(I-A)\,A^{i-1}x(1,k) + K_d\left[y^*(i+1) - y^*(i)\right] \qquad (15)$$

It can be observed that convergence $u(1,k) \to u^*(1)$ occurs first. After that the output convergence $y(2,k) \to y^*(2)$ is achieved. Then after one or few batches, $u(2,k) \to u^*(2)$ convergence is achieved followed by $y(3,k) \to y^*(3)$. Convergence of other inputs $u(i,k)$'s and corresponding outputs $y(i+1,k)$'s occurs in further batches. The batch to batch sequential convergence of samples is shown in Figure 1.

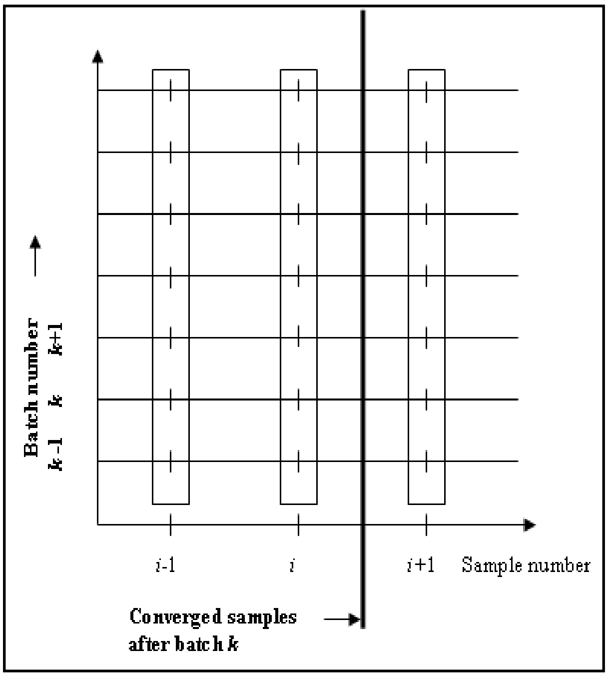

Figure 1 Sequential Convergence in ILC

2) Convergence Condition for Control Input Sequence

To obtain a stable and bounded sequence of control input $u(i,k)$ in (13), the Eigenvalues of $(I - K_dCB)$ must lie inside the unit circle. Hence, the maximum absolute eigenvalue should be less than unity as follows:

$$\max\left|\lambda\left(I - K_dCB\right)\right| < 1 \qquad (16)$$

For monotonic convergence, the condition in (16) is expressed as norm [21]:

$$\left\|I - K_dCB\right\| < 1 \qquad (17)$$

The (17) gives the necessary and sufficient condition for convergence of D-ILC algorithm. It is pointed that since this condition does not depend on the system matrix A, it marks the ability of ILC algorithm to achieve convergence even when the model parameters are unknown.

3) Rate of Convergence of Control Input Errors

Using D-ILC algorithm, the component-wise control input errors between desired control input $u^*(i)$ and $u(i,k+1)$ are calculated as follows:

$$\begin{aligned} u^*(i) - u(i,k+1) &= u^*(i) - \begin{bmatrix} u(i,k) \\ +K_d\{e(i+1,k) \\ -e(i,k)\} \end{bmatrix} \\ &= u^*(i) - u(i,k) \\ &\quad - K_dC\left[\begin{Bmatrix} (Ax^*(i) + Bu^*(i)) \\ -\left(Ax(i,k) + Bu(i,k)\right) \end{Bmatrix}\right] \\ &\quad + K_d\left\{y^*(i) - y(i,k)\right\} \end{aligned} \qquad (18)$$

Let's denote $u^*(i) - u(i,k) = \Delta u(i,k)$ and use in (18) and after simplification we get:

$$\Delta u(i,k+1) = \left(I - K_dCB\right)\left\{\Delta u(i,k)\right\} + K_dC\,(I-A)\left\{x^*(i) - x(i,k)\right\} \qquad (19)$$

Since initial conditions are preserved & initial error is zero, i.e. $x^*(1) = x(1,k)$, hence at 1st time index the error is:

$$\Delta u(1,k+1) = \left(I - K_dCB\right)\left\{\Delta u(1,k)\right\} \qquad (20)$$

At i^{th} time index, the error becomes:

$$\begin{aligned} \Delta u(i,k+1) &= \left(I - K_dCB\right)\Delta u(i,k) \\ &\quad + K_dC\,(I-A)\,B\Delta u(i-1,k) \\ &\quad + \cdots + K_dCA^{i-1}\,(I-A)\,B\Delta u(2,k) \\ &\quad + K_dCA^{i-2}\,(I-A)\,B\Delta u(1,k) \end{aligned} \qquad (21)$$

By writing (20) and (21) in matrix form, the evolution of vectors of control input errors from batch to batch is given as follows:

$$\Delta u(k+1) = T\ \Delta u(k) \qquad (22)$$

where, the operator matrix T controls the evolution of control input errors from batch to batch as follows:

$$\begin{bmatrix} (I - K_dCB) & 0 & 0 & \cdots & 0 \\ K_dC(I - A)B & (I - K_dCB) & 0 & \cdots & 0 \\ K_dCA(I - A)B & K_dC(I - A)B & \ddots & 0 & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ K_dCA^{M-2}(I - A)B & \cdots & \cdots & \cdots & (I - K_dCB) \end{bmatrix} \quad (23)$$

The solution of (22) is given as follows:

$$\Delta u(k) = T^{k-1} \Delta u(1) \quad (24)$$

Lemma:

The D-ILC algorithm produces a sequence of bounded control inputs which over the long term converge component-wise to the desired control input sequence at the rate equal to the magnitude of the Eigenvalue of the matrix relating the evolution of control input error provided initial conditions are same and desired output matches with the measured output at the beginning of each batch.

Proof of Lemma:

Consider the solution for evolution of control input errors in (24). Due to repeated eigenvalues, matrix T cannot be diagonalised using eigen-decomposition or SVD because there are repeated eigen vectors which make the matrix of Eigen vectors singular. Therefore, the following assumptions have been considered:

1. The rate of convergence of each component $\Delta u(i,k)$ can be found if $(M \times M)$ matrix T is decomposed into Jordan Normal form. For a non-singular matrix Q such that $Q^{-1} \times Q = I$, Jordan Decomposition is $T = Q^{-1}\, D\, Q$.

2. The matrix D has the eigenvalues of matrix T as its diagonal elements along with a sub-diagonal containing all 1's.

3. To calculate the convergence rates of individual components of control input error vector $\Delta u(k)$ in (24) we have to decompose as follows [22]:

$$T^{k-1} = Q^{-1}\, D^{k-1}\, Q \quad (25)$$

4. The control input error in (22) can be written as follows:

$$\Delta u(k) = T^{k-1} \Delta u(1) = Q^{-1} D^{k-1} Q\, \Delta u(1) \quad (26)$$

5. The rate of convergence $\Gamma_{\Delta u(i,k)}$ is the ratio of ∞-norm at batch k with respect to the ∞-norm at batch $k-1$:

$$\Gamma_{\Delta u(i,k)} = \frac{\left\| \Delta u(i,k) \right\|_\infty}{\left\| \Delta u(i,k-1) \right\|_\infty} = \frac{\left[\begin{array}{l} \left|\lambda^{k-1}\right| + \left|(k-1)\lambda^{k-2}\right| \\ + \left| \frac{(k-1)(k-2)}{2!} \lambda^{k-3} \right| \\ + \cdots + \left| \frac{(k-1)\ldots(k-i+1)}{(i-1)!} \lambda^{k-i} \right| \end{array} \right]}{\left[\begin{array}{l} \left|\lambda^{k-2}\right| + \left|(k-2)\lambda^{k-3}\right| \\ + \left| \frac{(k-2)(k-3)}{2!} \lambda^{k-4} \right| \\ + \cdots + \left| \frac{(k-2)\ldots(k-1-i+1)}{(i-1)!} \lambda^{k-1-i} \right| \end{array} \right]} \quad (27)$$

Consequently, after simplification, (27) can be written as follows:

$$\Gamma_{\Delta u(i,k)} = |\lambda| \sum_{j=0}^{i-1} \left\{ \left| \begin{pmatrix} k-1 \\ j \end{pmatrix} \right| \left| \left(\frac{1}{\lambda} \right)^j \right| \right\} \Big/ \left\{ \left| \begin{pmatrix} k-2 \\ j \end{pmatrix} \right| \left| \left(\frac{1}{\lambda} \right)^j \right| \right\} \quad (28)$$

where

$$\begin{pmatrix} k-1 \\ j \end{pmatrix} = \frac{(k-1)!}{(k-1-j))!\, j!} \quad (29)$$

In the limit $k \to \infty$, the ratio to the right of summation in (28) is unity, so that every component $\Delta u(i,k)$ has long term convergence rate $\approx |\lambda| = \left|(I - K_dCB)\right|$. □

III. CASE STUDY

A damped pendulum in Figure 2 is an interesting control problem which has been widely studied. Here we apply the D-type ILC algorithm for learning to track the desired angle & corresponding angular velocity by generating the desired control signal at each sample time. The set of desired control signal for a complete swing has been obtained from a fine-tuned PD controller. The angle θ is measured anti-clock-wise. The control input torque u is applied in anti-clock-wise direction as well. The pendulum is at rest with initial position $\theta_0 = \pi / 4$ radians at the left side. The initial angular velocity, $\omega_0 = 0$ rad/s. The pendulum has been simulated using following parameters:

Length L = 1m, mass m = 0.5 Kg, acceleration due to gravity g = 9.81 m/s2 and damping co-efficient b = 0.25

N-s/m. Using state vector $x = [\theta, \quad \omega]^T$, the linearised discrete-time state space matrices for the pendulum, sampled at 0.05 seconds are as follows:

$$A_d = \begin{bmatrix} 0.9879 & 0.0492 \\ -0.4824 & 0.9633 \end{bmatrix}, \quad B_d = \begin{bmatrix} 0.0012 \\ 0.0492 \end{bmatrix}, \quad C_d = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad D_d = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{30}$$

The time period for the pendulum is 2 seconds, so there are 40 samples for each swing from left to right and back.

For ILC update, the error considered is the state error. The selected learning gain matrix $K_d = 0.5 \times [0,1]$ allows the forward difference error of the angular velocity only to update the control inputs. The tracking of the desired control input u, angle θ and angular velocity ω has been monitored for 2500 swings in Figure 3 and Figure 4.

The RMS values reached minima around 2000 swings:

Final RMS of error in angle = 1.2015e-015,

Final RMS of error in angular velocity = 2.2210e-015,

Final RMS of control input error = 1.6876e-014

These values show that limit of precision has been reached. The evolution of the control input errors at selected time indices is shown in Figure 5. The semi-log plot in Figure 6 give the convergence rate of control input errors at selected time indices. The rate of convergence varies in the earlier swings. The convergence occurs sequentially as the control input for earlier time indices converge before the later ones. The kinks in convergence rates for initial swings occur due to oscillations or zero-crossings of the control input errors. Finally, the D-ILC algorithm achieves its limit at large batch numbers. As shown in Figure 6, that just before the control input errors are near the minimum threshold, all the individual components $\Delta u(i,k)$ achieve the same rate (0.9754) equal to the eigenvalue of matrix T.

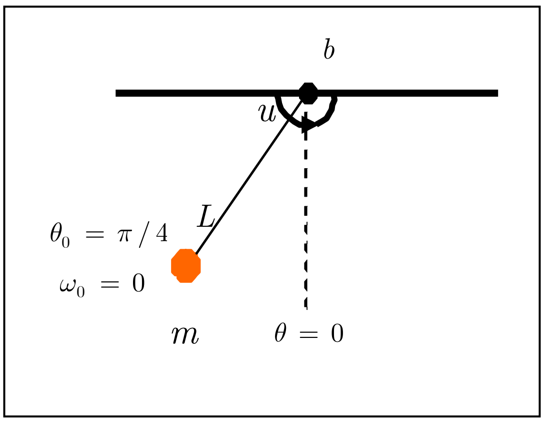

Figure 2 Damped Pendulum

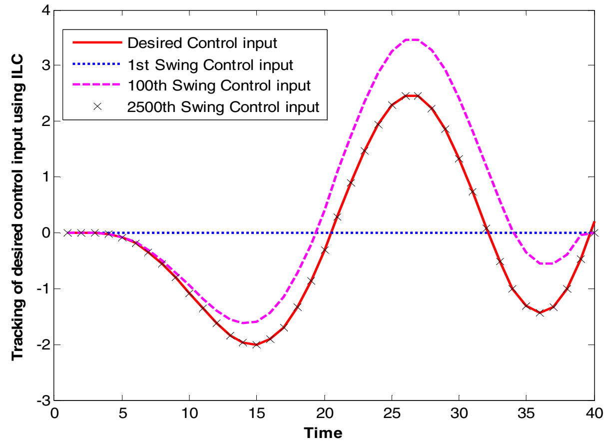

Figure 3 Tracking control input

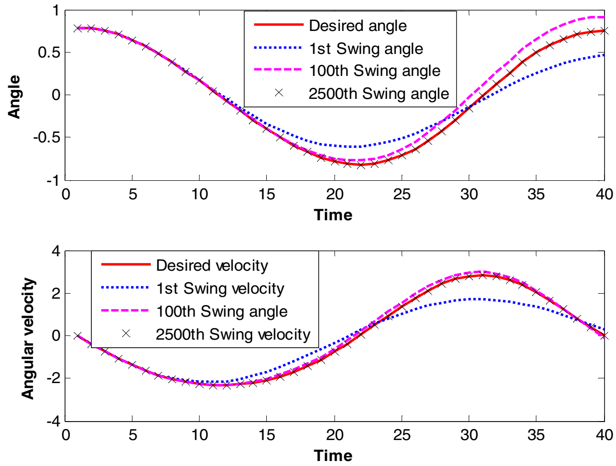

Figure 4 Tracking angle and velocity

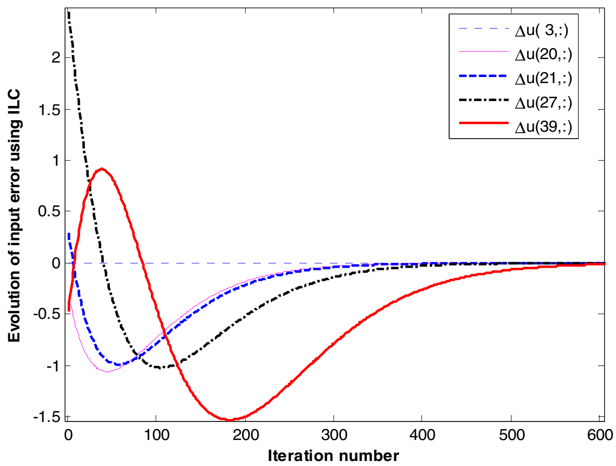

Figure 5 Evolution of control input errors

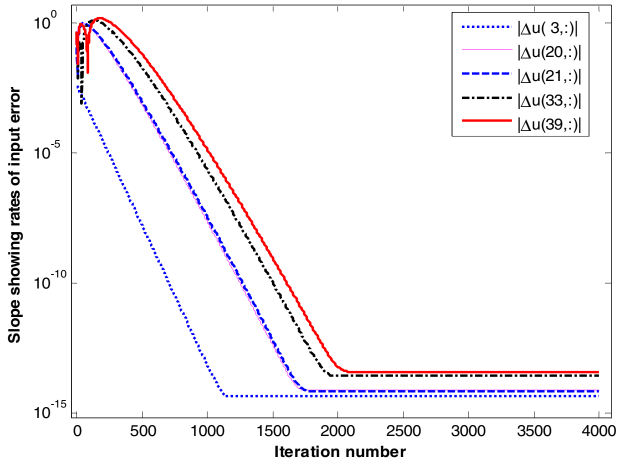

Figure 6 Convergence rate of control input errors

IV. CONCLUSION

The rate of convergence of D-ILC algorithm has been investigated in terms of input errors at component level which evolve sequentially. In the long term, all input components acquire same convergence rate equal to the eigenvalue of the Toeplitz matrix which relates the input errors from batch to batch. This knowledge can be helpful in designing the Iterative Learning controllers and their performance comparison.

REFERENCES

[1] Hyo-Sung, A., C. YangQuan, and K.L. Moore, *Iterative Learning Control: Brief Survey and Categorization.* Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, 2007. **37**(6): p. 1099-1121.

[2] UCHIYAMA, M., *Formation of high speed motion pattern of mechanical arm by trial.* . Transactions of the Society of Instrumentation and Control Engineers, 1978. **19**: p. 706-712.

[3] Craig, J.J. *Adaptive control of manipulators through repeated trials*. in *Proceedings of American Control Conference*. 1984. San Diego, CA,.

[4] Arimoto, S., S. Kawamura, and F. Miyazaki, *Bettering operation of Robots by learning.* Journal of Robotic Systems, 1984. **1**(2): p. 123-140.

[5] Kawamura, S., F. Miyazaki, and S. Arimoto. *Applications of learning method for dynamic control of robot manipulators*. in *24th IEEE Conference on Decision and Control*. 1985.

[6] Arimoto, S., S. Kawamura, and F. Miyazaki, *Convergence, stability and robustness of learning control schemes for robot manipulators*, in *Proceedings of the International Symposium on Robot Manipulators on Recent trends in robotics: modeling, control and education*. 1986, Elsevier North-Holland, Inc.: Albuquerque, New Mexico, United States. p. 307 - 316.

[7] Saab, S.S., *On the P-type learning control.* Automatic Control, IEEE Transactions on, 1994. **39**(11): p. 2298-2302.

[8] Owens, D.H. *Iterative learning control-convergence using high gain feedback*. in *Proceedings of the 31st IEEE Conference on Decision and Control*. 1992.

[9] Saab, S.S., *A discrete-time learning control algorithm for a class of linear time-invariant systems.* Automatic Control, IEEE Transactions on, 1995. **40**(6): p. 1138-1142.

[10] Samer, S.S., *Robustness and convergence rate of a discrete-timelearning control algorithm for a class of nonlinear systems.* International Journal of Robust and Nonlinear Control, 1999. **9**(9): p. 559-571.

[11] Arimoto, S., S. Kawamura, and F. Miyazaki. *Bettering operation of dynamic systems by learning: A new control theory for servomechanism or mechatronics systems*. in *The 23rd IEEE Conference on Decision and Control*. 1984.

[12] Bondi, P., G. Casalino, and L. Gambardella, *On the iterative learning control theory for robotic manipulators.* IEEE Journal of Robotics and Automation, 1988. **4**(1): p. 14-22.

[13] Wang, D., *On D-type and P-type ILC designs and anticipatory approach.* International Journal of Control, 2000. **73**(10): p. 890.

[14] Xu, J.-x. and Y. Tan, *On the P-type and Newton-type ILC schemes for dynamic systems with non-affine-in-input factors.* Automatica, 2002. **38**(7): p. 1237-1242.

[15] Shao-Juan, Y., W. Ju-Hua, and Y. Xue-Wen. *A PD-type open-closed-loop iterative learning control and its convergence for discrete systems*. in *Machine Learning and Cybernetics, 2002. Proceedings. 2002 International Conference on*. 2002.

[16] Zhang, B., G. Tang, and S. Zheng, *PD-type iterative learning control for nonlinear time-delay system with external disturbance.* Journal of Systems Engineering and Electronics, 2006. **17**(3): p. 600-605.

[17] YangQuan, C. and K.L. Moore. *PI-type iterative learning control revisited.* in *Proceedings of the American Control Conference*. 2002.

[18] Lequin, O., et al., *Iterative feedback tuning of PID parameters: comparison with classical tuning rules.* Control Engineering Practice, 2003. **11**(9): p. 1023-1033.

[19] Madady, A., *PID Type Iterative Learning Control with Optimal Gains.* International Journal of Control, Automation, and Systems, 2008. **6**(2): p. 194-203.

[20] Heinzinger, G., et al. *Robust learning control*. in *Proceedings of the 28th IEEE Conference on Decision and Control*. 1989.

[21] Yongqiang, Y., A. Tayebi, and L. Xiaoping. *A Unit-Gain D-type Iterative Learning Control Scheme: Application to a 6-DOF Robot Manipulator*. in *Intelligent Control, 2007. ISIC 2007. IEEE 22nd International Symposium on*. 2007.

[22] Galor, O., *Multi-Dimensional, First-Order, Linear Systems: Characterization*, in *Discrete Dynamical Systems*. 2007, Springer Berlin Heidelberg. p. 59-91.

[23] Nocedal, J. and S.J. Wright, *Numerical Optimization*, in *Springer Series in Operations Research and Financial Engineering*, T.V. Mikosch, S.I. Resnick, and S.M. Robinson, Editors. 2006, Springer: New York.