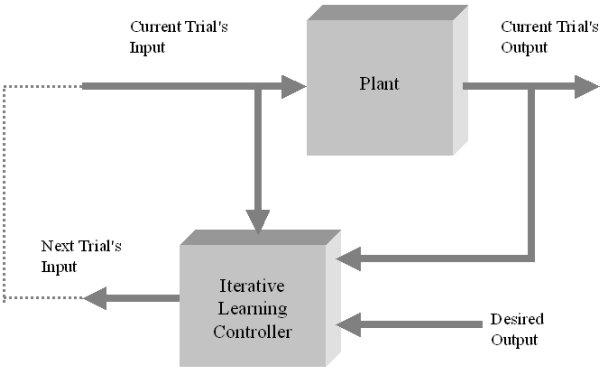
# An Introduction to Iterative Learning Control

**Kevin L. Moore**

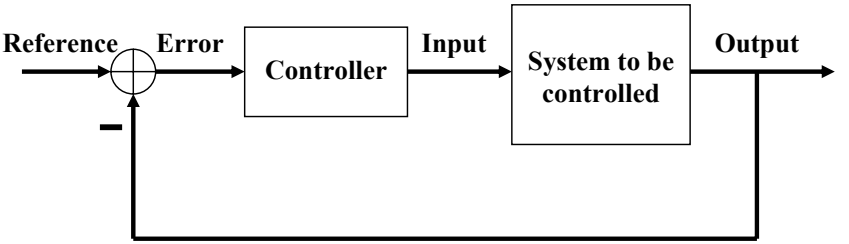
Center for Self-Organizing and Intelligent Systems
Dept. of Electrical and Computer Engineering
Utah State University



# Outline

- Introduction
- Control System Design: Motivation for ILC
- Iterative Learning Control: The Basic Idea
- A Short History of ILC
- ILC Problem Formulation
- LTI ILC: Operator-Theoretic Approach
- ILC as a 2-D Problem
- Multi-Loop Control Approach to ILC
- Higher-Order ILC: A Matrix Fraction Approach
- More on the Nature of the ILC Solution: Equivalence of ILC to One-Step Ahead Control
- Conclusion

# Control Design Problem



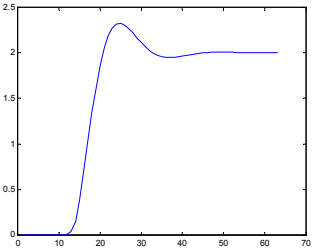

**Given:** System to be controlled.

**Find:** Controller (using feedback).

**Such that:** 1) Closed-loop system is stable.
2) Steady-state error is acceptable.
3) Transient response is acceptable.

# Motivation for the Problem of Iterative Learning Control

- Transient response design is hard:

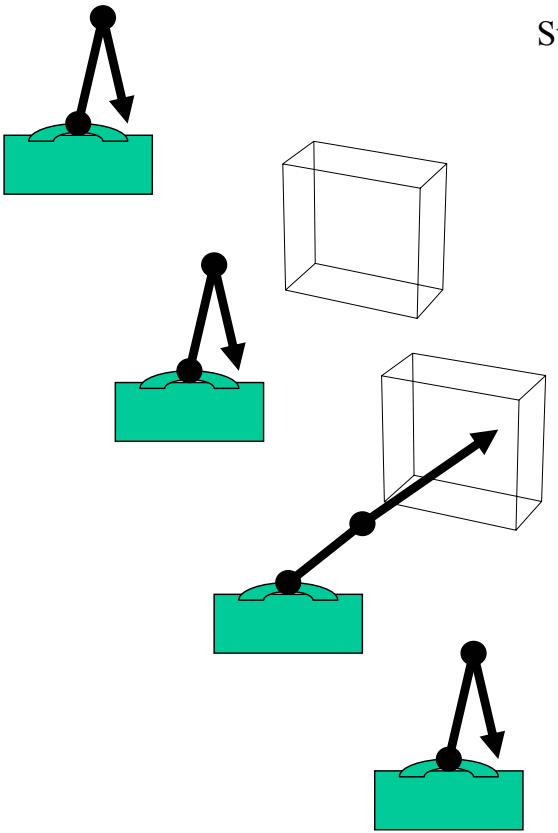



  1) Robustness is always an issue:
     - Modelling uncertainty.
     - Parameter variations.
     - Disturbances.
  2) Lack of theory (design uncertainty):
     - Relation between pole/zero locations and transient response.
     - Relation between Q/R weighting matrices in optimal control and transient response.
     - Nonlinear systems.

- Many systems of interest in applications are operated in a repetitive fashion.

- Iterative Learning Control (ILC) is a methodology that tries to address the problem of transient response performance for systems that operate repetitively.

# Systems that Execute the Same Trajectory Repetitively



Step 1: Robot at rest, waiting for workpiece.

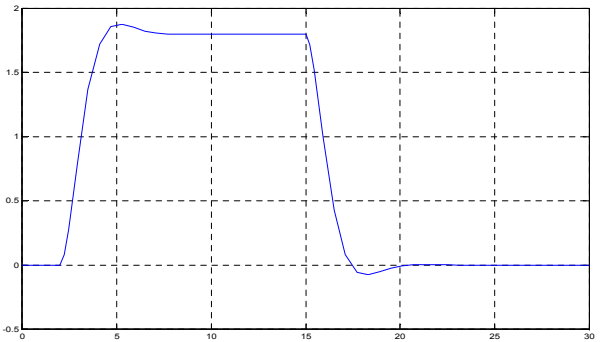
Step 2: Workpiece moved into position.

Step 3: Robot moves to desired location

Step 4: Robot returns to rest and waits for next workpiece.

# Errors are Repeated When Trajectories are Repeated

•A typical joint angle trajectory for the example might look like this:

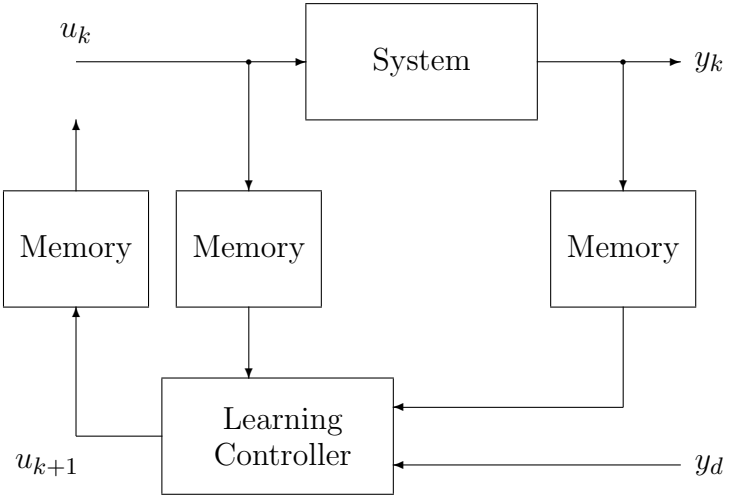



•Each time the system is operated it will see the same overshoot, rise time, settling time, and steady-state error.

•Iterative learning control attempts to *improve the transient response* by *adjusting the input to the plant during future* system operation based on the *errors observed during past* operation.

# Iterative Learning Control - 1

- Standard iterative learning control scheme:



# Iterative Learning Control - 2

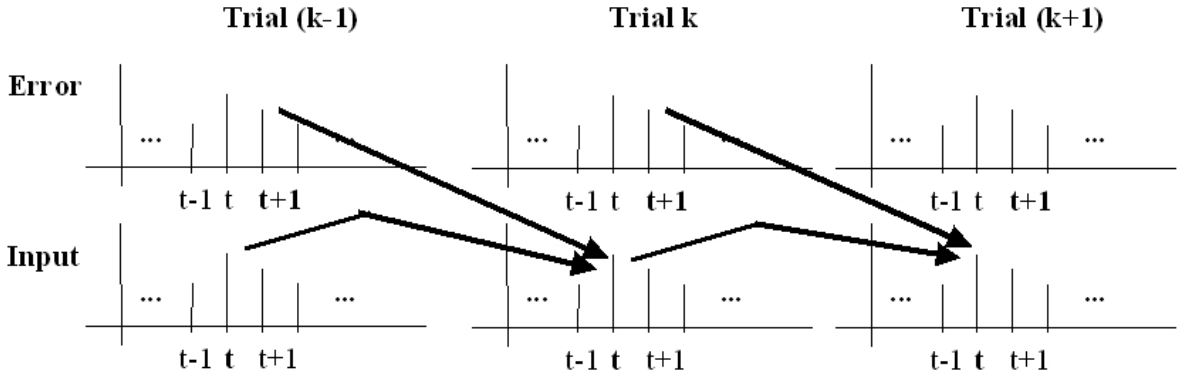
- A typical ILC algorithm has the form:

$$u_{k+1}(t) = u_k(t) + f(e_k(t+1))$$

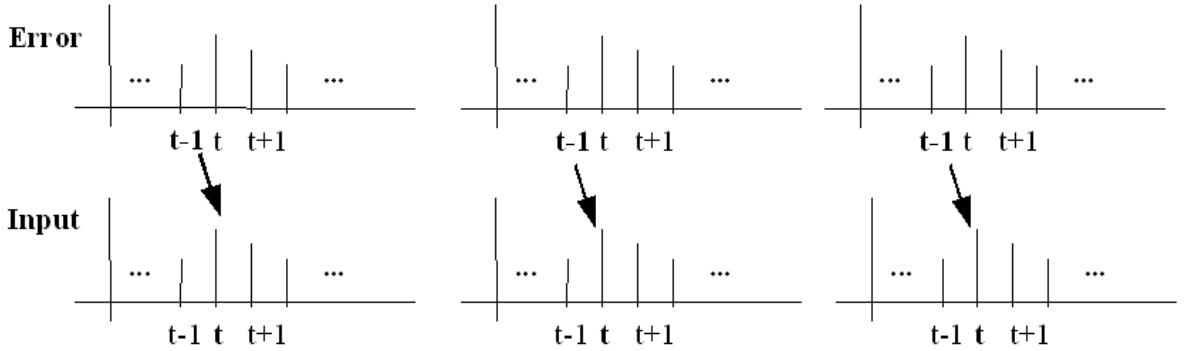
- Conventional feedback control has the form:

$$u_{k+1}(t) = f(e_{k+1}(t-1))$$

- Standard ILC assumptions include:
  - Stable dynamics or some kind of Lipschitz condition.
  - System returns to the same initial conditions at the start of each trial.
  - Each trial has the same length.




(a) ILC: $u_{k+1}(t) = u_k(t) + f(e_k(t+1))$



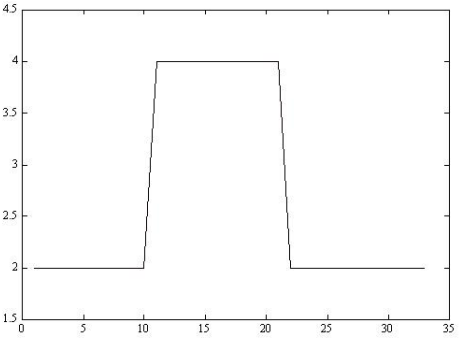

(b) Conventional feedback: $u_{k+1}(t) = f(e_{k+1}(t-1))$

# A Simple Linear Example - 1

- Consider the plant:

$$\begin{aligned} y(t+1) &= -.7y(t) - .012y(t-1) + u(t) \\ y(0) &= 2 \\ y(1) &= 2 \end{aligned}$$

- We wish to force the system to follow a signal $y_d$:

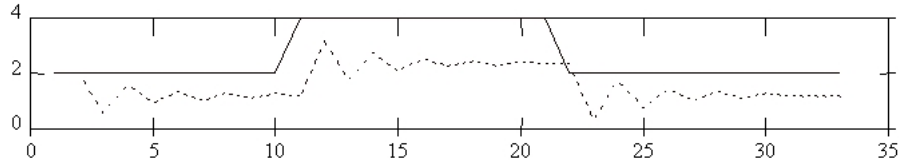


# A Simple Linear Example - 2

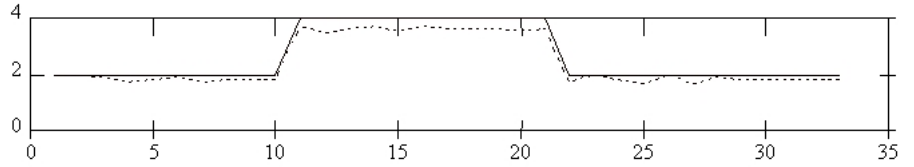
- Use the following ILC procedure:

  1. Let
  $$u_0(t) = y_d(t)$$
  2. Run the system
  3. Compute
  $$e_0(t) = y_d(t) - y_0(t)$$
  4. Let
  $$u_1(t) = u_0(t) + 0.5e_0(t+1)$$
  5. Iterate

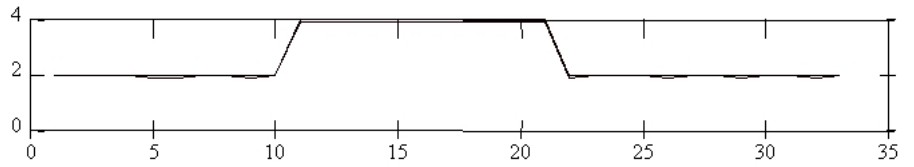
- Each iteration shows an improvement in tracking performance (plot shows desired and actual output on first, 5th, and 10th trials and input on 10th trial).


4
2
0
0
5
10
15
20
25
30
35

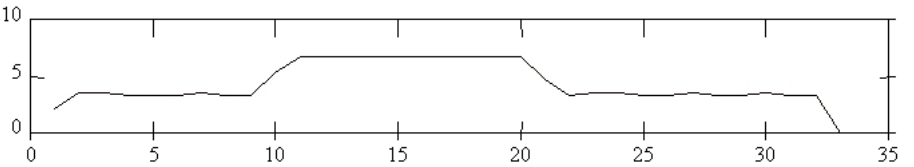
(a)


4
2
0
0
5
10
15
20
25
30
35

(b)


4
2
0
0
5
10
15
20
25
30
35

(c)


10
5
0
0
5
10
15
20
25
30
35

(d)

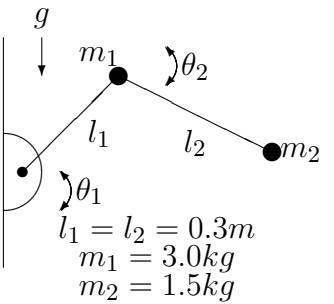
# Adaptive ILC for a Robotic Manipulator - 1

- Consider a simple two-link manipulator modelled by:

$$A(x_k)\ddot{x}_k + B(x_k, \dot{x}_k)\dot{x}_k + C(x_k) = u_k$$

where

$$\begin{aligned}
x(t) &= (\theta_1(t), \theta_2(t))^T \\
A(x) &= \begin{pmatrix} .54 + .27\cos\theta_2 & .135 + .135\cos\theta_2 \\ .135 + .135\cos\theta_2 & .135 \end{pmatrix} \\
B(x, \dot{x}) &= \begin{pmatrix} .135\sin\theta_2 & 0 \\ -.27\sin\theta_2 & -.135(\sin\theta_2)\dot{\theta}_2 \end{pmatrix} \\
C(x) &= \begin{pmatrix} 13.1625\sin\theta_1 + 4.3875\sin(\theta_1 + \theta_2) \\ 4.3875\sin(\theta_1 + \theta_2) \end{pmatrix} \\
u_k(t) &= \text{vector of torques applied to the joints}
\end{aligned}$$



# Adaptive ILC for a Robotic Manipulator - 2

- Define the vectors:

$$\begin{aligned} y_k &= (x_k^T, \dot{x}_k^T, \ddot{x}_k^T)^T \\ y_d &= (x_d^T, \dot{x}_d^T, \ddot{x}_d^T)^T \end{aligned}$$

- The learning controller is defined by:

$$\begin{aligned} u_k &= r_k - \alpha_k \Gamma y_k + C(x_d(0)) \\ r_{k+1} &= r_k + \alpha_k \Gamma e_k \\ \alpha_{k+1} &= \alpha_k + \gamma \|e_k\|^m \end{aligned}$$

- $\Gamma$ is a fixed feedback gain matrix that has been made time-varying through the multiplication by the gain $\alpha_k$.
- $r_k$ can be described as a time-varying reference input. $r_k(t)$ and adaptation of $\alpha_k$ are effectively the ILC part of the algorithm.
- With this algorithm we have combined conventional feedback with iterative learning control.

# Adaptive ILC for a Robotic Manipulator - 3

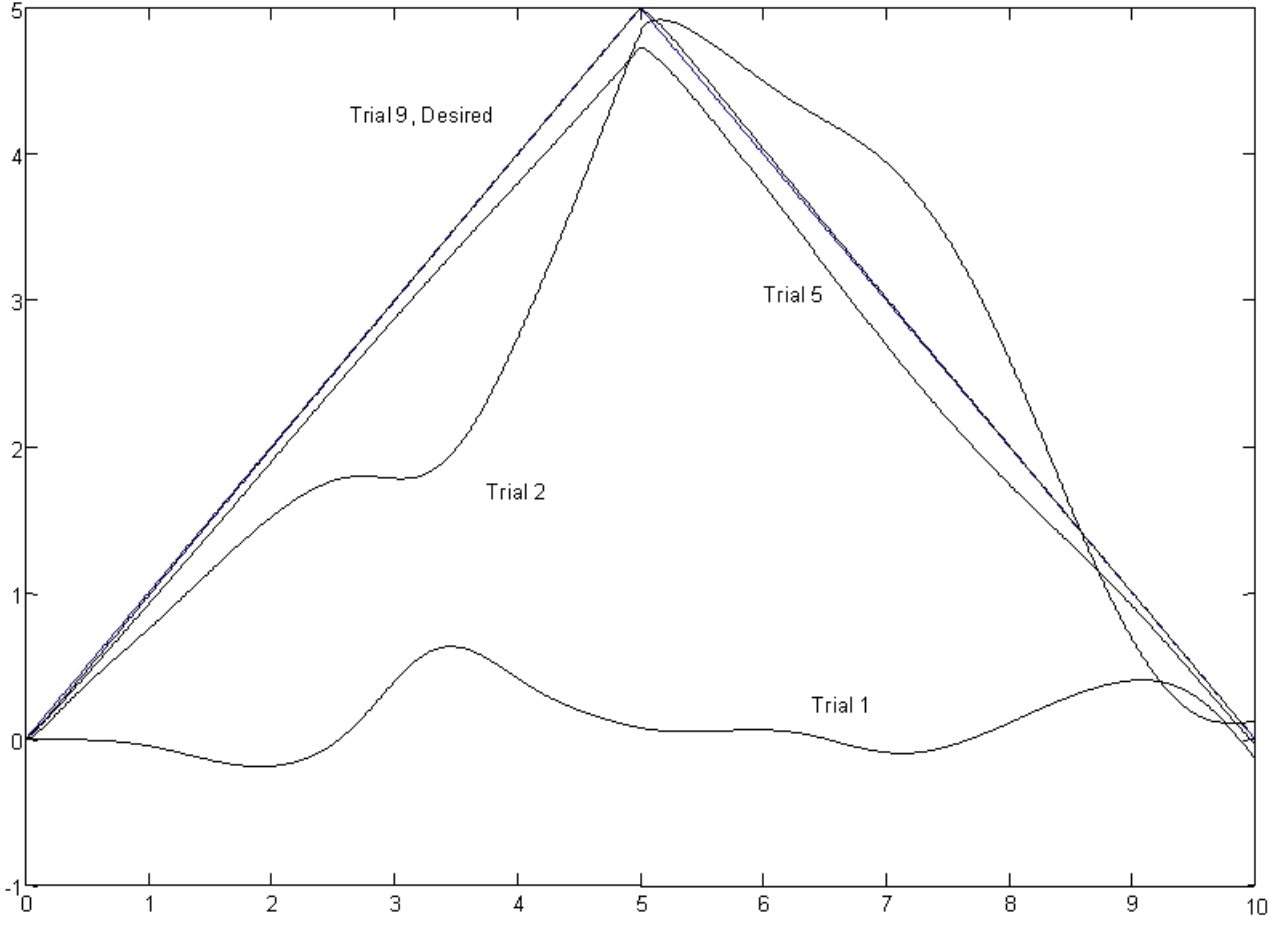
- The basic idea of the algorithm:
  - Make $\alpha_k$ larger at each trial by adding a positive number that is linked to the norm of the error.
  - When the algorithm begins to converge, the gain $\alpha_k$ will eventually stop growing.
  - The convergence proof depends on a high-gain feedback result.
- To simulate this ILC algorithm for the two-joint manipulator described above we used first-order highpass filters of the form

$$H(s) = \frac{10s}{s+10}$$

  to estimate the joint accelerations $\ddot{\theta}_1$ and $\ddot{\theta}_2$ from the joint velocities $\dot{\theta}_1$ and $\dot{\theta}_2$, respectively.
- The gain matrix $\Gamma = [P\ L\ K]$ used in the feedback and learning control law was defined by:

$$P = \begin{pmatrix} 50.0 & 0 \\ 0 & 50.0 \end{pmatrix} \quad L = \begin{pmatrix} 65.0 & 0 \\ 0 & 65.0 \end{pmatrix} \quad K = \begin{pmatrix} 2.5 & 0 \\ 0 & 2.5 \end{pmatrix}$$

- The adaptive gain adjustment in the learning controller uses $\gamma = .1$ and $\alpha_0$ is initialized to 0.01.
- Each iteration shows an improvement in tracking performance.


Trial 9, Desired
Trial 5
Trial 2
Trial 1
5
4
3
2
1
0
-1
0
1
2
3
4
5
6
7
8
9
10

# A Short Overview of ILC - 1

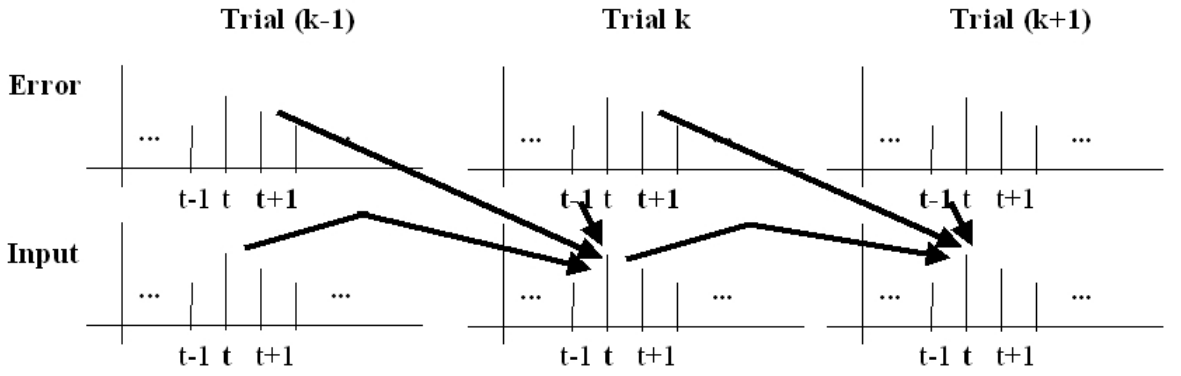
- *Historical Roots* of ILC go back about twenty-five years:
  - Idea of a "multipass" system studied by Owens and Rogers in mid- to late-1970's, with several resulting monographs.
  - Learning control concept introduced (in Japanese) by Uchiyama in 1978.
  - Pioneering work of Arimoto, et al. 1984-present.
  - Related research in repetitive and periodic control.
  - 1993 Springer monograph had about 90 ILC references.
  - 1997 Asian Control Conference had 30 papers on ILC (out of 600 papers presented at the meeting) and the first panel session on this topic.
  - 1998 survey paper has about 250 ILC references.
  - Web-based online, searchable bibliographic database maintained by Yangquan Chen has about 500 references (see http://cicserver.ee.nus.edu.sg/ ilc).
  - ILC Workshop and Roundtable and three devoted sessions at 1998 CDC.
  - Edited book by Bien and Xu resulting from 1997 ASCC.
  - At least four Ph.D. dissertations on ILC since 1998.
  - Springer monograph by Chen and Wen, 1999.
  - Special sessions at 2000 ASCC, ICARV 2000, and 2nd Int. Conference on nD Systems.
  - Tutorial at ICARV 200 and first CDC Tutorial Workshop; first IFAC World Congress special session.
  - ILC Summer School at Utah State University, June 2003.

# A Short Overview of ILC - 2

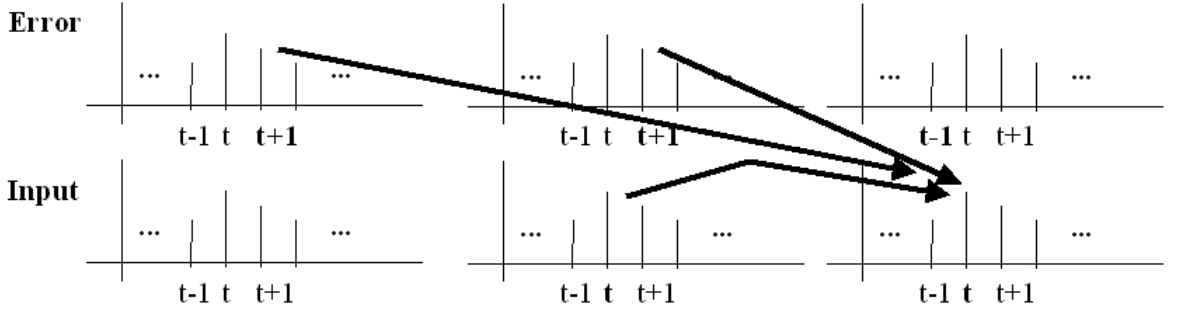
- *Past* work in the field demonstrated the usefulness and applicability of the concept of ILC:
  - Linear systems.
  - Classes of nonlinear systems.
  - Applications to robotic systems.
- *Present* status of the field reflects the continuing efforts of researchers to:
  - Develop design tools.
  - Extend earlier results to broader classes of systems.
  - Realize a wider range of applications.
  - Understand and interpret ILC in terms of other control paradigms and in the larger context of learning in general.

# A Partial Classification of ILC Research

- Systems:
  - Open-loop vs. closed-loop.
  - Discrete-time vs. continuous-time.
  - Linear vs. nonlinear.
  - Time-invariant or time-varying.
  - Relative degree 1 vs. higher relative degree.
  - Same initial state vs. variable initial state.
  - Presence of disturbances.
- Update algorithm:
  - Linear ILC vs. nonlinear ILC.
  - First-order ILC vs. higher-order.
  - Current cycle vs. past cycle.
  - Fixed ILC or adaptive ILC.
  - Time-domain vs. frequency analysis.
  - Analysis vs. design.
  - Assumptions on plant knowledge.
- Applications: robotics, chemical processing, mechatronic systems.



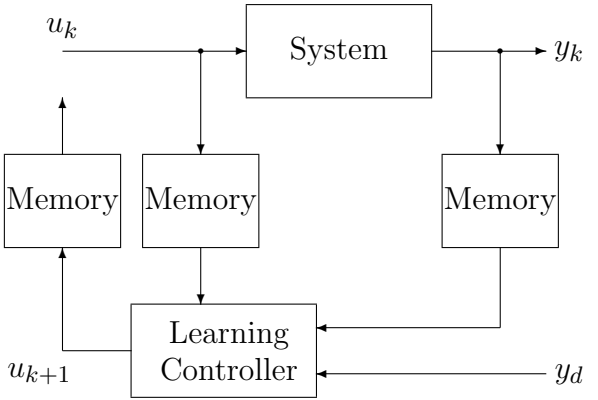

(a) ILC with Current Cycle Feedback




(b) Higher-Order ILC

# ILC Problem Formulation - 1

- Standard iterative learning control scheme:




- **Goal**: Find a learning control algorithm

$$u_{k+1}(t) = f_L(u_k(t); y_k(t), y_d(t))$$

so that for all $t \in [0, t_f]$

$$\lim_{k\to\infty} y_k(t) = y_d(t)$$

# ILC Problem Formulation - 2

- Given
  - a system $S$ (possibly nonlinear, time-varying), with
    $$y_k(t) = f_{S_k}(u_k(t))$$
  - a desired response $y_d(t)$, defined on the interval $(t_0, t_f)$.
- Find
  - a system $L$ (possibly unknown, non-linear, time-varying).
- So that
  - the sequence of inputs produced by the iteration
    $$u_{k+1}(t) = f_{L_k}(u_k, y_k, y_d) = f_{L_k}(u_k, f_{S_k}(u_k), y_d)$$
    converges to a fixed point $u^*$.
- Such that
  $$\lim_{k\to\infty} \|y_d - y_k\| = \|y_d - T_s u^*\|$$
  is minimum on the interval $(t_0, t_f)$ over all possible systems $L$, for a specified norm.
- **Linear case**: $u_{k+1} = T_u u_k + T_e(y_d - y_k)$

# Some Learning Control Algorithms

- Arimoto first proposed a learning control algorithm of the form:

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

  Convergence is assured if $\|I - CB\Gamma\|_i < 1$.

- Arimoto has also considered more general algorithms of the form:

$$u_{k+1} = u_k + \Phi e_k + \Gamma \dot{e}_k + \Psi \int e_k dt$$

- Various researchers have used gradient methods to optimize the gain $G_k$ in:

$$u_{k+1}(t) = u_k(t) + G_k e_k(t+1)$$

- It is also useful for design to specify the learning control algorithm in the frequency domain, for example:

$$U_{k+1}(s) = L(s)[U_k(s) + aE_k(s)]$$

- Many schemes in the literature can be classified with one of the algorithms given above.

# LTI ILC Convergence Conditions - 1

- **Theorem**: For the plant $y_k = T_s u_k$, the linear time-invariant learning control algorithm

$$u_{k+1} = T_u u_k + T_e(y_d - y_k)$$

converges to a fixed point $u^*(t)$ given by

$$u^*(t) = (I - T_u + T_e T_s)^{-1} T_e y_d(t)$$

with a final error

$$e^*(t) = \lim_{k \to \infty} (y_k - y_d) = (I - T_s(I - T_u + T_e T_s)^{-1} T_e) y_d(t)$$

defined on the interval $(t_0, t_f)$ if

$$\|T_u - T_e T_s\|_i < 1$$

- **Observation**:
  - If $T_u = I$ then $\|e^*(t)\| = 0$ for all $t \in [t_o, t_f]$.
  - Otherwise the error will be non-zero.

# LTI ILC Convergence Conditions - 2

- General linear algorithm:

$$u_{k+1}(t) = T_u u_k(t) + T_e(y_d(t) - y_k(t))$$

- Can show that, if

(i) $T_u = I$ and $\|I - T_e T_s\|_i < 1$, then

$$e^*(t) = \lim_{k\to\infty}(y_d(t) - y_k(t)) = 0$$

(ii) $T_u \neq I$ and $\|T_u - T_e T_s\|_i < 1$, then

$$e^*(t) = (I - T_s(I - T_u + T_e T_s)^{-1} T_e) y_d(t) \neq 0$$

- Note: The convergence condition for (i) can require system invertibility.

**e.g.**, For convergence in the space of finite-energy signals, with a requirement that our ILC filters be causal, (i) holds only for those systems whose transfer functions are non-minimum phase and have no zeros at infinity.

# LTI ILC - Convergence with Zero Error

- Let $T_u = I$ to get $e^*(t) = 0$ over the entire interval.
- Condition for convergence becomes:

$$\|I - H_e H_s\|_i < 1$$

- Unfortunately, this can be overly restrictive.
- **Example**:

  If $u \in \mathbf{U} = \mathbf{L}_2^r(0, \infty)$ and $y \in \mathbf{Y} = \mathbf{L}_2^m(0, \infty)$ then the condition for convergence is:

$$\|I - H_e(s) H_s(s)\|_\infty < 1$$

  However, we can easily show:

  **Theorem**: There exists a proper, stable, LTI system $H_e(s) \in \mathbf{H}_\infty$ such that

$$\|I - H_e(s) H_s(s)\|_\infty < 1$$

  if and only if $H_s(s)$ is invertible over the space $\mathbf{H}_\infty$.

# LTI Learning Control - Nature of the Solution

- **Question**: Given $T_s$, how do we pick $T_u$ and $T_e$ to make the final error $e^*(t)$ as "small" as possible, for the general linear ILC algorithm:

$$u_{k+1}(t) = T_u u_k(t) + T_e(y_d(t) - y_k(t))$$

- **Answer**: Let $T_n^*$ solve the problem:

$$\min_{T_n} \|(I - T_s T_n) y_d\|$$

It turns out that we can specify $T_u$ and $T_e$ in terms of $T_n^*$ and the resulting learning controller converges to an optimal system input given by:

$$u^*(t) = T_n^* y_d(t)$$

- **Conclusion**:The essential effect of a properly designed learning controller is to produce the output of the best possible inverse of the system in the direction of $y_d$.

# LTI ILC - Convergence with Non-Zero Error

- **OPT1**: Let $u_k \in \mathbf{U},\ y_d, y_k \in \mathbf{Y}$ and $T_s, T_u, T_e \in \mathbf{X}$. Then given $y_d$ and $T_s$, find $T_u^*$ and $T_e^*$ that solve

$$\min_{T_u, T_e \in \mathbf{X}} \|(I - T_s(I - T_u + T_e T_s)^{-1} T_e) y_d\|$$

  subject to $\|T_u - T_e T_s\|_i < 1$.

- **OPT2**: Let $y_d \in \mathbf{Y}$ and let $T_n, T_s \in \mathbf{X}$. Then given $y_d$ and $T_s$, find $T_n^*$ that solve

$$\min_{T_n \in \mathbf{X}} \|(I - T_s T_n) y_d\|$$

- **Theorem**: Let $T_n^*$ be the solution of OPT2. Factor $T_n^* = T_m^* T_e^*$ where $T_m^{*^{-1}} \in \mathbf{X}$ and $\|I - T_m^{*^{-1}}\|_i < 1$. Define $T_u^* = I - T_m^{*^{-1}} + T_e^* T_s$. Then $T_u^*$ and $T_e^*$ are the solution of OPT1.

- If we plug these into the expression for the fixed-point of the input to the system we find:

$$u^*(t) = T_n^* y_d(t)$$

- Note: The factorization in the Theorem can always be done, with the result that $u^*(t) = T_n^* y_d(t)$.

# Interlude: The Two-Dimensional Nature of ILC - 1

- Let the plant be a scalar (and possibly time-varying), discrete-time dynamical system, described as:

$$y_k(t+1) \quad = \quad f_S[y_k(t), u_k(t), t]$$

  Here:

  - $k$ denotes a trial (or execution, repetition, pass, etc.).
  - $t \in [0, N]$ denotes time (integer-valued).
  - $y_k(0) = y_d(0) = y_0$ for all $k$.

- Use a general form of a typical ILC algorithm for a system with relative degree one:

$$u_{k+1}(t) \quad = \quad f_L[u_k(t), e_k(t+1), k]$$

  where

  - $e_k(t) = y_d(t) - y_k(t)$ is the error on trial $k$.
  - $y_d(t)$ is a desired output signal.

# Interlude: The Two-Dimensional Nature of ILC - 2

- Combine the plant equation with the ILC update rule to get:

$$\begin{aligned} y_{k+1}(t+1) &= f_S[y_k(t), u_{k+1}(t), t] \\ &= f_S[f_L[y_k(t), u_k(t), e_k(t+1), k], t] \end{aligned}$$

- By changing the notation slightly we get:

$$y(k+1, t+1) = f[y_k(t), u(k,t), e(k,t+1), k, t]$$

- Clearly this is a 2-D system:
  - Dynamic equation indexed by two variables: $k$ and $t$.
  - $k$ defines the repetition domain (Longman/Phan terminology); $t$ is the normal time-domain variable.
- Nevertheless, there are two basic points by which ILC differs from a complete 2-D system design problem:
  - One of the dimensions (time) is a finite, fixed interval, thus convergence in that direction (traditional stability) is always assured for linear systems.
  - In the ILC problem we admit non-causal processing in one dimension (time) but not in the other (repetition).

# A Representation of Discrete-Time Linear ILC - 1

- Consider a discrete-time plant of the form:

$$Y(z) = H(z)U(z) = (h_m z^{-m} + h_{m+1} z^{-(m+1)} + h_{m+2} z^{-(m+2)} + \cdots)U(z)$$

- Define (for $m = 1$):

$$\begin{aligned} U_k &= [u_k(0), u_k(1), \cdots, u_k(N-1)]^T \\ Y_k &= [y_k(1), y_k(2), \cdots, y_k(N)]^T \\ Y_d &= [y_d(1), y_d(2), \cdots, y_d(N)]^T \end{aligned}$$

- Thus the linear plant can be described by $Y_k = HU_k$ where (assuming relative degree one):

$$H = \begin{bmatrix} h_1 & 0 & 0 & \dots & 0 \\ h_2 & h_1 & 0 & \dots & 0 \\ h_3 & h_2 & h_1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_N & h_{N-1} & h_{N-2} & \dots & h_1 \end{bmatrix}$$

# A Representation of Discrete-Time Linear ILC - 2

- For the linear, time-varying case, suppose we have the plant given by:

$$\begin{aligned} x_k(t+1) &= A(t)x_k(t) + B(t)u_k(t) \\ y_k(t) &= C(t)x_k(t) + D(t)u_k(t) \\ x_k(0) &= x_0 \end{aligned}$$

  Then the same notation results in $Y_k = HU_k$, where now:

$$H = \begin{bmatrix} h_{m,0} & 0 & 0 & \dots & 0 \\ h_{m+1,0} & h_{m,1} & 0 & \dots & 0 \\ h_{m+2,0} & h_{m+1,1} & h_{m,2} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{m+N-1,0} & h_{m+N-2,1} & h_{m+N-3,2} & \dots & h_{m,N-1} \end{bmatrix}$$

- Clearly this approach allows us to represent our multi-pass (Owens and Rogers terminology) dynamical system in $R^1$ into a static system in $R^N$.

# A Representation of Discrete-Time Linear ILC - 3

- Suppose we have a simple ILC update equation in our $R^1$ representation:

$$u_{k+1}(t) = u_k(t) + \gamma e_k(t+1)$$

for $t \in [0, N]$, and where $\gamma$ is a constant gain. In our $R^N$ representation, we write:

$$U_{k+1} = U_k + \Gamma E_k$$

where

$$\Gamma = \text{diag}(\gamma)$$

- Suppose we filter during an ILC update, such as the following update equation in our $R^1$ representation:

$$u_{k+1}(t) = u_k(t) + L(z)e_k(t+1)$$

Then our $R^N$ representation would have the form:

$$U_{k+1} = U_k + LE_k$$

where, if L was time-invariant and causal, then it would have the lower-triangular form:

$$L = \begin{bmatrix} L_m & 0 & 0 & \dots & 0 \\ L_{m+1} & L_m & 0 & \dots & 0 \\ L_{m+2} & L_{m+1} & L_m & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ L_{m+N-1} & L_{m+N-2} & L_{m+N-3} & \dots & L_m \end{bmatrix}$$

# A Representation of Discrete-Time Linear ILC - 4

- We may similarly consider time-varying and noncausal filters in the ILC update law:

$$U_{k+1} = U_k + LE_k$$

- A causal (in time), time-varying filter in the ILC update law might look like, for example:

$$L = \begin{bmatrix} n_{1,0} & 0 & 0 & \dots & 0 \\ n_{2,0} & n_{1,1} & 0 & \dots & 0 \\ n_{3,0} & n_{2,1} & n_{1,2} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ n_{N,0} & n_{N-1,1} & n_{N-2,2} & \dots & n_{1,N-1} \end{bmatrix}$$

- A non-causal (in time), time-invariant averaging filter in the ILC update law might look like, for example:

$$L = \begin{bmatrix} K & K & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & K & K & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & K & K & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & K & K & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & K & K \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & K \end{bmatrix}$$

- These issues have been described in some detail by Longman.

# Multi-Loop Approach to ILC - 1

- Suppose we use the discrete-time version of Arimoto's ILC algorithm (assume relative degree $m = 1$):

$$u_{k+1}(t) = u_k(t) + \gamma e_k(t+1) = u_k + \gamma(y_d - Hu_k)$$

- Then the individual components of $u_{k+1}$ are:

$$\begin{aligned}
u_{k+1}(0) &= u_k(0) + \gamma(y_d(1) - h_1 u_k(0)) \\
u_{k+1}(1) &= u_k(1) + \gamma(y_d(2) - h_1 u_k(1) - h_2 u_k(0)) \\
&\vdots \\
u_{k+1}(t) &= u_k(t) + \gamma(y_d(t+1) - h_1 u_k(t) - \sum_{n=0}^{t-1} h_{t+1-n} u_k(n)) \\
&\vdots
\end{aligned}$$

# Multi-Loop Approach to ILC - 2

- Introduce a new shift variable, $w$, with the property that:

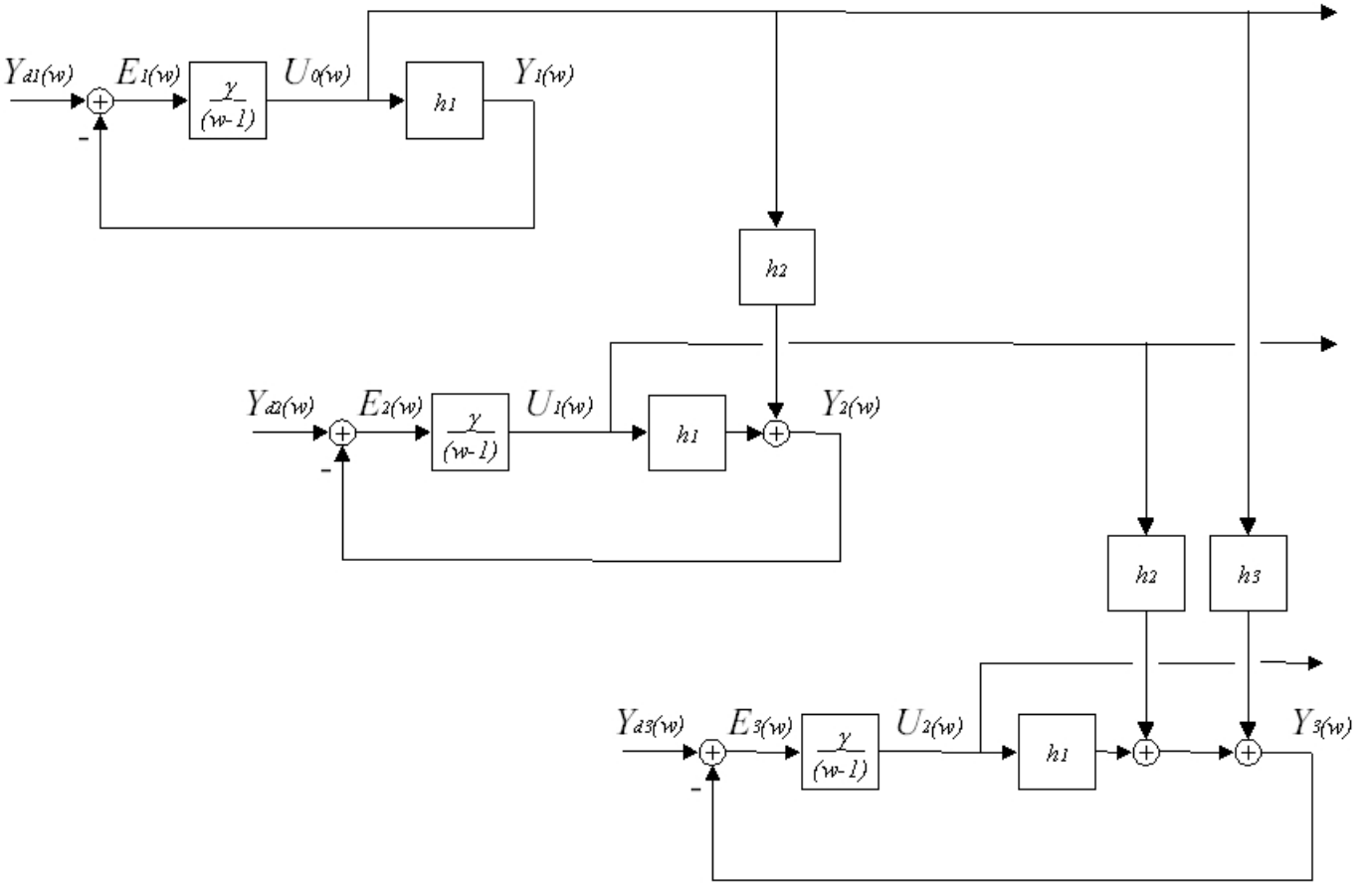
$$w^{-1}u_k(t) = u_{k-1}(t)$$

- The individual components of the $w$-transform of $u_{k+1}(t)$, denoted $U_t(w)$ are:

$$\begin{aligned}
wU_0(w) &= U_0(w) + \gamma(Yd_1(w) - h_1U_0(w)) \\
wU_1(w) &= U_1(w) + \gamma(Yd_2(w) - h_1U_1(w) - h_2U_0(w)) \\
&\vdots \\
wU_t(w) &= U_t(w) + \gamma(Yd_{t+1}(w) - h_1U_t(w) - \sum_{n=0}^{t-1} h_{t+1-n}U_n(w) \\
&\vdots
\end{aligned}$$

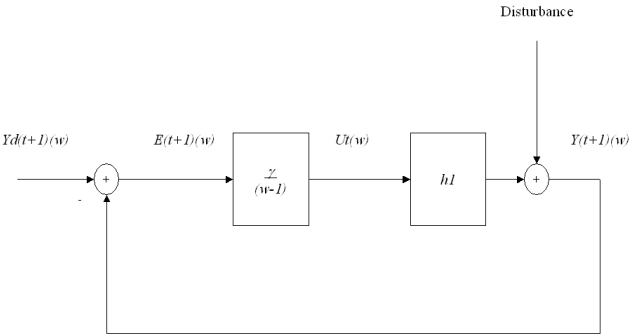
- Note that:

$$U_t(w) = \frac{\gamma}{(w-1)}E_{t+1}(w)$$


Yd1(w)
E1(w)
γ
(w-1)
U0(w)
h1
Y1(w)
-
h2
Yd2(w)
E2(w)
γ
(w-1)
U1(w)
h1
Y2(w)
-
h2
h3
Yd3(w)
E3(w)
γ
(w-1)
U2(w)
h1
Y3(w)
-

# Multi-Loop Interpretation of ILC

- The ILC scheme can thus be interpreted as a multi-loop control system in the $w$-domain.
- At each time step the closed-loop system (relative to trials) has a simple feedback structure.




- From basic systems theory it is clear that we have convergence if and only if the pole of this closed-loop system lies inside the unit disk, or, equivalently:

$$|1 - \gamma h_1| < 1$$

- Note that with suitable choice of norm this becomes equivalent to Arimoto's convergence condition indicated above for systems with relative degree one (because $h_1 = CB$).

# Extensions of the Multi-Loop Approach

(1) ILC Algorithm Design

- This multi-loop interpretation suggests a new ILC algorithm (reported at 1998 CDC):

  1. Apply two different inputs at the same time step, $u_k(0)$ and $u_{k+1}(0)$, to the system, using a fixed gain $\gamma$ of any value and keeping all other inputs fixed. Then:

$$h_1 = \frac{1 - e_{k+1}(1)/e_k(1)}{\gamma}$$

  2. Then pick a new $\gamma$ to give deadbeat convergence:

$$\gamma_{\text{new}} = \frac{1}{h_1}$$

  3. Compute (after the second pass):

$$h_i = -\frac{e_{k+1}(i) - e_k(i)}{u_{k+1}(0) - u_k(0)}$$

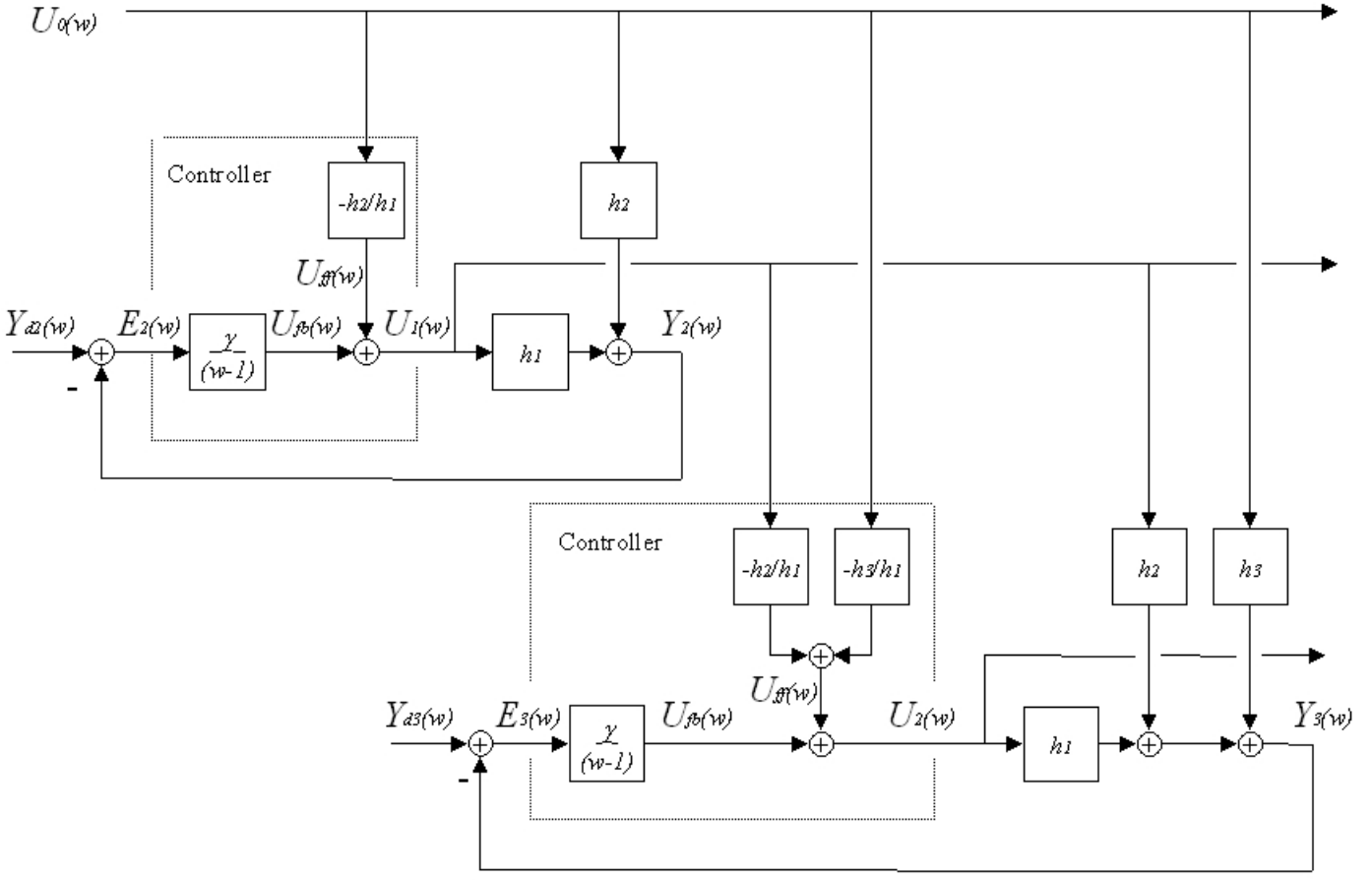
# Extensions (cont.)

ILC Algorithm Design (cont.)

4. Exploit the diagonal structure of the multi-loop control system and implement an uncoupled feedforward approach to the input update for the third trial:

$$\begin{aligned} u_{k+1}^{fb}(t) &= u_k^{fb}(t) + \gamma e_k(t+1) \\ u_{k+1}^{ff}(t) &= -\frac{1}{h_1}\sum_{n=0}^{t-1} h_{t+1-n} u_{k+1}(n) \\ u_{k+1}(t) &= u_{k+1}^{fb}(t) + u_{k+1}^{ff}(t) \end{aligned}$$

5. The system will demonstrate exact convergence on the fourth trial (the deadbeat pole location provides convergence one step (trial) after the feedforward correction has been applied).


$U_{0(w)}$
Controller
$-h_2/h_1$
$h_2$
$U_{ff(w)}$
$Y_{d2(w)}$
$E_{2(w)}$
$\frac{Y}{(w-1)}$
$U_{fb(w)}$
$U_{1(w)}$
$h_1$
$Y_{2(w)}$
-
Controller
$-h_2/h_1$
$-h_3/h_1$
$h_2$
$h_3$
$U_{ff(w)}$
$Y_{d3(w)}$
$E_{3(w)}$
$\frac{Y}{(w-1)}$
$U_{fb(w)}$
$U_{2(w)}$
$h_1$
$Y_{3(w)}$
-

# Extensions (cont.)

(2) Adaptive Gain Adjustment

- In the ILC algorithm presented above we computed the gain according to $\gamma_{\text{new}} = 1/h_1$, where $h_1$ was computed by:

$$h_1 = \frac{1 - e_{k+1}(1)/e_k(1)}{\gamma}$$

- In general this will not be a very robust approach.
- Alternately, treat this as a control problem where we have an unknown (sign and magnitude) static plant with a time-varying disturbance that we wish to force to track a desired output.
- One way to solve such a problem is with a sliding mode controller (one could put a different sliding mode control law at each time step).

(3) Estimation of $h_1$

- For systems with measurement noise we can use parameter estimation techniques to estimate $h_1$ (reported at '99 CDC).

(4) Multivariable Systems

- Results can be extended to MIMO systems (reported at recent ICARV2000).

# Extensions (cont.)

(5) Nonlinear and Time-Varying Systems:

- Consider the affine nonlinear system:

$$\begin{aligned} x_k(t+1) &= f(x_k(t)) + g(x_k(t))u_k(t) \\ y_k(t) &= Cx_k(t) \end{aligned}$$

with the ILC algorithm $u_{k+1}(t) = u_k(t) + \gamma e_k(t+1)$. Then we can derive the (previously reported in the literature) convergence condition:

$$\max_x |1 - \gamma C g(x)| < 1$$

- Similarly, for the time-varying system:

$$\begin{aligned} x_k(t+1) &= f(x_k(t), t) + g(x_k(t), t)u_k(t) \\ y_k(t) &= Cx_k(t) \end{aligned}$$

with the time-varying ILC algorithm $u_{k+1}(t) = u_k(t) + \gamma_t e_k(t+1)$ the multi-loop interpretation allows us to derive the convergence condition:

$$\max_x |1 - \gamma_t C g(x,t)| < 1, \qquad t = 0, 1, \cdots, t_f$$

Using the same approach, with the time-invariant ILC rule $u_{k+1}(t) = u_k(t) + \gamma e_k(t+1)$ we get:

$$\max_{x,t} |1 - \gamma C g(x,t)| < 1$$

# Extensions (cont).

(6) Higher-Order ILC:

- The term $\gamma/(w-1)$ is effectively the controller of the system (in the repetition domain).
- This can be replaced with a more general expression $C(w)$.
- For example, a "higher-order" ILC algorithm could have the form:

$$u_{k+1}(t) = k_1 u_k(t) + k_2 u_{k-1}(t) + \gamma e_k(t+1)$$

  which corresponds to:

$$C(w) = \frac{\gamma w}{w^2 - k_1 w - k_2}$$

- It has been suggested in the literature that such schemes can give faster convergence.
- We see this may be due to more freedom in placing the poles (in the $w$-plane).
- However, we have shown dead-beat control using the zero-order ILC scheme. Thus, higher-order ILC is not required to speed up convergence.
- But, there can be a benefit to the higher-order ILC schemes:
  - $C(w)$ can implement a Kalman filter/parameter estimator to determine the Markov parameter $h_1$ and $E\{y(t)\}$ when the system is subject to noise.
  - $C(w)$ can be used to implement a robust controller in the repetition domain.
  - A matrix fraction approach to ILC filter design can be developed from these ideas (presented at 2nd Int. Workshop on nD Systems).

# Repetition-Domain Frequency Representation of Higher-Order ILC: Filtering in the Repetition Domain

- The generalized higher-order ILC algorithm has been considered in the time-domain by various researchers (see Phan, Longman, and Moore):

$$\begin{aligned} U_{k+1} &= \bar{D}_n U_k + \bar{D}_{n-1} U_{k-1} + \cdots + \bar{D}_1 U_{k-n+1} + \bar{D}_0 U_{k-n} + \\ &\qquad N_n E_k + N_{n-1} E_{k-1} + \cdots + N_1 E_{k-n+1} + N_0 E_{k-n} \end{aligned}$$

- Here, we apply the shift variable, $w$ introduced above, to get $\bar{D}_c(w)U(w) = N_c(w)E(w)$, where

$$\begin{aligned} \bar{D}_c(w) &= Iw^{n+1} - \bar{D}_{n-1}w^n - \cdots - \bar{D}_1 w - \bar{D}_0 \\ N_c(w) &= N_n w^n + N_{n-1}w^{n-1} + \cdots + N_1 w + N_0 \end{aligned}$$

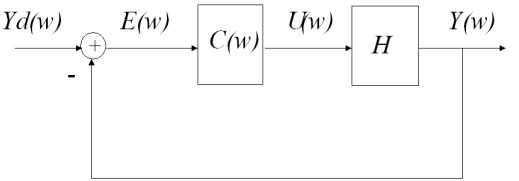
  This can be written in a matrix fraction as $U(w) = C(w)E(w)$, where $C(w) = \bar{D}_c^{-1}(w)N_c(w)$.

- Thus, through the addition of higher-order terms in the update algorithm, the ILC problem has been converted from a static multivariable representation to a dynamic (in the repetition domain) multivariable representation.

- Note that we will always get a linear, time-invariant system like this, even if the actual plant is time-varying or affine nonlinear.

- Also because $\bar{D}_c(w)$ is of degree $n+1$ and $N_c(w)$ is of degree $n$, we have relative degree one in the repetition-domain.

# Convergence and Error Analysis




Convergence

- From the figure we see that in the repetition-domain the closed-loop dynamics are defined by:

$$\begin{aligned} G_{cl}(w) &= H[I + C(w)H]^{-1}C(w) \\ &= H[(w-1)\bar{D}_c(w) + N_c(w)H]^{-1}N_c(w) \end{aligned}$$

- Thus the ILC algorithm will converge (i.e., $E_k \rightarrow$ a constant) if $G_{cl}$ is stable.
- Determining the stability of this feedback system may not be trivial:
  - It is a multivariable feedback system of dimension $N$, where $N$ could be very large.
- But, the problem may be simplified due to the fact that the plant $H$ is a constant, lower-triangular matrix.

# Convergence and Error Analysis (cont.)

Convergence with Zero Error

- Because $Y_d$ is a constant and our "plant" is type zero (e.g., $H$ is a constant matrix), the internal model principle applied in the repetition domain requires that $C(w)$ should have an integrator effect to cause $E_k \to 0$.
- Thus, we modify the ILC update algorithm as:

$$\begin{aligned} U_{k+1} &= (I - D_{n-1})U_k + (D_{n-1} - D_{n-2})U_{k-1} + \cdots \\ &\quad +(D_2 - D_1)U_{k-n+2} + (D_1 - D_0)U_{k-n+1} + D_0 U_{k-n} \\ &\quad +N_n E_k + N_{n-1}E_{k-1} + \cdots + N_1 E_{k-n+1} + N_0 E_{k-n} \end{aligned}$$

- Taking the "w-transform" of the ILC update equation, combining terms, and simplifying gives:

$$(w-1)D_c(w)U(w) = N_c(w)E(w)$$

where

$$\begin{aligned} D_c(w) &= w^n + D_{n-1}w^{n-1} + \cdots + D_1 w + D_0 \\ N_c(w) &= N_n w^n + N_{n-1}w^{n-1} + \cdots + N_1 w + N_0 \end{aligned}$$

# Convergence and Error Analysis (cont.)

Convergence with Zero Error (cont.)

- This can also be written in a matrix fraction as:

$$U(w) = C(w)E(w)$$

  but where we now have:

$$C(w) \quad = \quad (w-1)^{-1} D_c^{-1}(w) N_c(w)$$

- Thus, we now have an integrator in the feedback loop (a discrete integrator, in the repetition domain) and, applying the final value theorem to $G_{cl}$, we get $E_k \to 0$ as long as the ILC algorithm converges (i.e., as long as $G_{cl}$ is stable).

# ILC and One-Step-Ahead Minimum Error Control Input Sequence Resulting from ILC

- As above, let the plant to be controlled be of the form:

$$Y(z) = H(z)U(z) = (h_1 z^{-1} + h_2 z^{-2} + h_3 z^{-3} + \cdots)U(z)$$

and let:

$$\begin{aligned} u_k &= [u_k(0), u_k(1), \cdots, u_k(N-1)]^T \\ y_k &= [y_k(1), y_k(2), \cdots, y_k(N)]^T \\ y_d &= [y_d(1), y_d(2), \cdots, y_d(N)]^T \end{aligned}$$

so that we can write:

$$y_k = H u_k$$

where $H$ is the matrix of the system's Markov parameters.

# ILC and One-Step-Ahead Minimum Error Control Input Sequence Resulting from ILC (cont.)

- Then using a standard Arimoto-type ILC update equation we can compute the final (steady-state in trial) values of each component (in time) of the input sequence. For each $t$ let:

$$u^*(t) = \lim_{k \to \infty} u_k(t)$$

Then:

$$\begin{aligned} u^*(0) &= \frac{1}{h_1} y_d(1) \\ u^*(1) &= \frac{1}{h_1} [y_d(2) - h_2 u^*(0)] \\ u^*(2) &= \frac{1}{h_1} [y_d(3) - h_2 u^*(1) - h_3 u^*(0)] \\ &\vdots \end{aligned}$$

# ILC and One-Step-Ahead Minimum Error Control Input Sequence Resulting from One-Step-Ahead Predictor

- Consider the transfer function form of the plant:

$$Y(z) = H(z)U_{OSA}(z) = z^{-1}\frac{b_0 + b_1 z^{-1} + \cdots + b_n z^{-n}}{1 + a_1 z^{-1} + \cdots + a_n z^{-n}} U_{OSA}(z)$$

- Using a one-step-ahead minimum error predictor to minimize:

$$J_1(t+1) = \frac{1}{2}[y(t+1) - y_d(t+1)]^2$$

  produces the input defined by:

$$\begin{aligned} u_{OSA}(t) = \frac{1}{b_0}[&y_d(t+1) + a_1 y(t) + a_2 y(t-1) \\ &+ \cdots + a_n y(t-n+1) \\ &- b_1 u_{OSA}(t-1) - b_2 u_{OSA}(t-2) \\ &- \cdots - b_n u_{OSA}(t-n)] \end{aligned}$$

- This feedback control law results in (for all $t \geq 1$):

$$y(t) = y_d(t)$$

# ILC and One-Step-Ahead Minimum Error Control Relationship Between $u^*(t)$ and $u_{OSA}(t)$

- Carry out a long-division process on the plant to compute find:

$$\begin{aligned} h_1 &= b_0 \\ h_2 &= b_1 - a_1 b_0 \\ &\quad b_1 - a_1 h_1 \\ h_3 &= b_2 - a_2 b_0 - a_1(b_1 - a_1 b_0) \\ &\quad b_2 - a_2 h_1 - a_1 h_2 \end{aligned}$$

- Consider the expressions for $u_{OSA}(t)$ for $t = 0, 1, 2, \ldots$, assuming:
  - All initial conditions on the input are equal to zero for $t < 0$.
  - All initial conditions on the output are zero for $t \leq 0$.
  - $y_d(0) = y(0) = 0$.

*For* $t = 0$ (using the fact that $b_0 = h_1$):

$$\begin{aligned} u_{OSA}(0) &= \frac{1}{b_0} y_d(1) \\ &= \frac{1}{h_1} y_d(1) \\ &= u^*(0) \end{aligned}$$

*For $t = 1$*: We can write $u_{OSA}(1) = \frac{1}{b_0}[y_d(2) + a_1 y(1) - b_1 u_{OSA}(0)]$. But, because we can express the plant output in terms of the Markov parameters, we have $y(1) = h_1 u_{OSA}(0)$. Combining this with the fact that $b_0 = h_1$ and that (as just derived above) $u_{OSA}(0) = u^*(0)$, we have

$$\begin{aligned} u_{OSA}(1) &= \frac{1}{h_1}[y_d(2) + a_1 h_1 u^*(0) - b_1 u^*(0)] \\ &= \frac{1}{h_1}[y_d(2) - (b_1 - a_1 h_1) u^*(0)] \\ &= \frac{1}{h_1}[y_d(2) - h_2 u^*(0)] \\ &= u^*(1) \end{aligned}$$

*For $t = 2$*: Similarly, using the additional fact that $y(2) = h_2 u_{OSA}(0) + h_1 u_{OSA}(1)$, it is easy to compute:

$$\begin{aligned} u_{OSA}(2) &= \frac{1}{b_0}[y_d(3) + a_1 y(2) + a_2 y(1) - b_1 u_{OSA}(1) - b_2 u_{OSA}(0)] \\ &= \frac{1}{b_0}[y_d(3) - a_1(h_2 u_{OSA}(0) + h_1 u_{OSA}(1)) + a_2 h_1 u_{OSA}(0) \\ &\qquad -b_1 u_{OSA}(1) - b_2 u_{OSA}(0)] \\ &= \frac{1}{b_0}[y_d(3) - (b_1 - a_1 h_1) u_{OSA}(1) - (b_2 - a_2 h_1 - a_1 h_2) u_{OSA}(0)] \\ &= \frac{1}{h_1}[y_d(3) - (b_1 - a_1 h_1) u^*(1) - (b_2 - a_2 h_1 - a_1 h_2) u^*(0)] \\ &= \frac{1}{h_1}[y_d(3) - h_2 u^*(1) - h_3 u^*(0)] \\ &= u^*(2) \end{aligned}$$

# Relationship Between $u^*(t)$ and $u_{OSA}(t)$ (cont.)

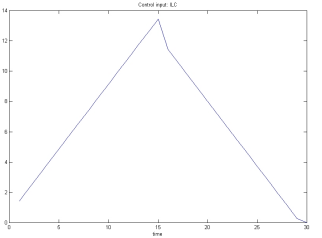
- The key fact is that, for all $t$:

$$u^*(t) = u_{OSA}(t)$$

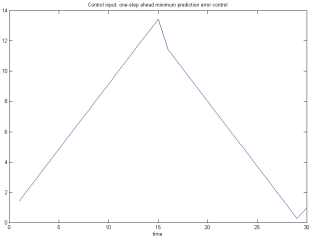
- Recall that the one-step-ahead minimum prediction error controller effectively inverts the plant (after discounting the time delay).
- Thus, we have reinforced the concept that the ILC technique effectively inverts the plant, as noted above.
- The one-step-ahead minimum prediction error controller is a feedback controller.
- The ILC technique is an open-loop control scheme that converges to the same input sequence as the one-step-ahead minimum prediction error controller.
- Control energy considerations:
  - A common objection to ILC is that it is often applied without regard to control energy.
  - This same objection applies to the one-step-ahead minimum prediction error controller.
  - However, the weighted one-step-ahead controller can be introduced to minimize the cost function
$$J_1(t+1) = \frac{1}{2}[y(t+1) - y_d(t+1)]^2 + \frac{\lambda}{2}u_{OSA}^2(t)$$
  - This concept can also be applied to ILC design.

# Illustration

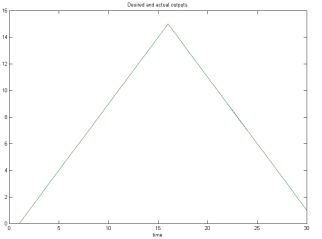
- Plant: $y_k(t+1) = 0.4y_k(t) + 0.7u_k(t)$
- Resulting ILC input sequence ($u^*(t)$):




- Resulting one-step-ahead minimum error predictor output($u_{OSA}(t)$):




- Actual and desired outputs from each control technique:



# Concluding Comments - 1

- ILC is a control methodology that can be applied to improve the transient response of systems that operate repetitively.

- An operator-theoretic analysis shows that the essential effect of an ILC scheme is to produce the output of the best possible inverse of the system.

- We noted that ILC is fundamentally a special case of a two-dimensional system.

- A multi-loop control interpretation of the signal flow in ILC can be used to:
  - Derive convergence conditions.
  - Develop an iterative learning control algorithm that can converge in as few as four trials.
  - Apply ILC to the case when the plant to be controlled is subject to measurement noise.
  - Apply ILC to nonlinear and time-varying systems.
  - Develop an adaptive gain adjustment ILC technique.

# Concluding Comments - 2

- We have presented a matrix fraction approach to iterative learning control:
  - We showed how "filtering" in the repetition domain can be interpreted as transforming the two-dimensional system design problem into a one-dimensional multivariable design problem.
  - Conditions for ILC convergence in terms of the ILC gain filters were derived using a matrix fraction approach and the final value theorem.

- We have shown that the control input resulting from a convergent ILC scheme is identical to that resulting from a standard one-step-ahead controller acting to produce minimum prediction error when there is full plant knowledge:
  - This result reinforces the notion that the effect of an ILC system is to invert the plant.
  - This result shows that the ILC process, which is fundamentally open-loop in time, is equivalent (in its final result) to a closed-loop control system

# Related References


1. "Iterative Learning Control for Multivariable Systems with an Application to Mobile Robot Path Tracking Control," Kevin L. Moore and Vikas Bahl, in *Proceedings of the 2000 International Conference on Automation, Robotics, and Control*, Singapore, December 2000.

2. "A Non-Standard Iterative Learning Control Approach to Tracking Periodic Signals in Discrete-Time Nonlinear Systems," Kevin L. Moore, *International Journal of Control*, Vol. 73, No. 10, 955-967, July 2000.

3. "On the Relationship Between Iterative Learning Control and One-Step-Ahead Minimum Prediction Error Control," Kevin L. Moore, in *Proceedings of the 3rd Asian Control Conference*, p. 1861-1865, Shanghai, China, July 2000.

4. "A Matrix-Fraction Approach to Higher-Order Iterative Learning Control: 2-D Dynamics through Repetition-Domain Filtering," Kevin L. Moore, in *Proceedings of 2nd International Workshop on Multidimensional (nD) Systems*, pp. 99-104, Lower Selesia, Poland, June 2000.

5. "Unified Formulation of Linear Iterative Learning Control," Minh Q. Phan, Richard W. Longman, and Kevin L. Moore, in *Proceedings of AAS/AIAA Flight Mechanics Meeting*, Clearwater Florida, January 2000.

# Related References (cont.)


6. "An Iterative learning Control Algorithm for Systems with Measurement Noise," Kevin L. Moore, in *Proceedings of the 1999 Conference on Decision and Control*, pp. 270-275Phoenix, AZ, Dec. 1999.

7. "Iterative Learning Control - An Expository Overview," Kevin L. Moore, invited paper in *Applied and Computational Controls, Signal Processing, and Circuits*, vol. 1, pp. 151-214, 1999.

8. "Multi-Loop Control Approach to Designing Iterative Learning Controllers," Kevin L. Moore, in *Proceedings of the 37th IEEE Conference on Decision and Control*, pp. 666-671, Tampa, FL, December 1998.

9. *Iterative Learning Control for Deterministic Systems*, Springer-Verlag Series on Advances in Industrial Control, Springer-Verlag, London, January 1993.

10. "Iterative Learning Control: A Survey and New Results," Kevin L. Moore, Mohammed Dahleh, and S.P. Bhattacharyya, *Journal of Robotic Systems*, vol. 9, no. 5, pp. 563-594, July 1992.